FACULTAD DE INGENIERIA U.N.A M.
DIVISION DE EDUCACION CONTINUA

CURSOS INSTITUCIONALES

SECRETARIA DE COMUNICACIONES Y TRANSPORTES

DBASE IV AVANZADO

Del 15 al 26 de agosto de 1994.

MATERIAL DIDACTICO

Ing. MERRY SAMPERIO

México, D.F.

CAPITULO UNO

Definición de necesidades

El primer paso en el desarrollo de una aplicación de dBASE IV es la definición de las necesidades del sistema. Si se está reemplazando un sistema manual ya existente por uno informatizado, el punto lógico de partida para resumir las funciones del nuevo software es una descripción detallada de los procedimientos manuales, acompañada de un conjunto de ejemplos de los formatos de los documentos que se utilizan.

Los usuarios desearán añadir nuevas posibilidades no incluidas en el sistema manual actual; esta "lista de deseos" llegará a ser una parte importante de las especificaciones del sistema.

En algunos casos, el sistema manual no existe, como cuando un negocio se expande en una nueva dirección, cuando una organización decide entrar en el mundo del proceso de datos, anteriormente proporcionado por un servicio de procesamiento de datos, o cuando los administradores desean utilizar la computadora para imprimir informes analíticos que son demasiado difíciles de producir a mano o que consumen demasiado tiempo.

Para definir las necesidades, se especifican detalladamente los siguientes procesos:

La entrada al sistema.	La información que deberá ser introducida en la computadora.
Las funciones del proceso de datos.	Los cálculos que la computadora debe realizar y la forma en que la información será transferida entre los archivos de la base de datos.
La salida del sistema.	Los formatos en que los datos serán visualizados en la pantalla o impresos.

Este capítulo describe un método sistemático para recoger la información necesaria para diseñar un sistema de bases de datos dBASE IV. Si ya ha implementado aplicaciones en otro lenguaje de alto nivel, muchos de los conceptos presentados aquí

Para cada nuevo informe, debe solicitar información que responda a una serie de preguntas cruciales al igual que en el caso de informes ya existentes:

- ¿Con qué frecuencia se imprimirán los informes?
- ¿Quién utilizará el informe?
- ¿Qué información específica debe incluirse?
- Para un informe en columnas, ¿en qué orden se deben imprimir las columnas?
- Para un informe de página llena, ¿en qué parte de la página debe colocarse cada elemento?
- ¿En qué orden deben imprimirse los registros?
- ¿Qué cálculos deben realizarse en cada registro?
- ¿Qué subtotales, totales y estadísticas se requieren?
- ¿Qué criterio determinará los registros de los informes?

Algunas veces, la forma más fácil de obtener esta información, sin formular preguntas en términos técnicos, es hacer un diseño preliminar burdo de una página típica para cada nuevo informe, incluyendo algunos datos representativos.

Evaluación de la lista de campos

Cuando haya listado todos los datos que el sistema debe contener, así como los informes que debe generar, vuelva a examinar la lista para tener la certeza de que son necesarios todos los campos. El estudio de los informes presentes y propuestos es útil para no caer en una trampa común: crear una base de datos inmensa con muchos campos de información. Es una pérdida de tiempo introducir información que ya existe si lo único que se pretende es ver los datos en la pantalla o imprimirlos en un formato similar.

Un campo pertenece a una base de datos sólo si cumple al menos uno de los siguientes criterios:

- ¿Existe la necesidad de ver la información en la pantalla?
- ¿El dato debe imprimirse en un informe?
- ¿Es necesaria la información para algún cálculo o estadística?
- ¿Será utilizado para seleccionar registros?
- ¿Será utilizado para especificar el orden en que debe visualizarse o imprimirse los registros?

Se debe tener especial cuidado si una base de datos necesita aparentemente gran cantidad de texto o campos numéricos que no van a ser utilizados en cálculos posteriores. Por ejemplo, debería evitar introducir en la base de datos una amplia descripción acerca de la actividad empresarial de cada cliente, si esta información está disponible en notas escritas a mano. Sin embargo, sería deseable incluir varios campos con códigos identificativos de la actividad del cliente para poder realizar una búsqueda rápida en un archivo y referirnos a las notas para más detalle.

En otros casos, decidir la información que va a ser incluida puede resultar menos claro. Supongamos que va a a realizar una base de datos para un médico que pide a todos sus pacientes nuevos que rellenen un cuestionario de cuatro hojas sobre su historial médico. La primera inclinación podría ser incluir todas las respuestas en una base de datos de historiales de pacientes, pero dependiendo de la utilización que se quiera hacer de la información, este esfuerzo puede resultar innecesario.

Por ejemplo, el historial médico puede incluir cuestiones sobre alergias. Para detectar si un paciente en concreto tiene la fiebre del heno, sería tan fácil extraer el folleto del paciente del archivo manual existente y hacer referencia al cuestionario original como si se hubiera recuperado el registro del paciente de la base de datos. Si, por el contrario, el médico desea saber cuántos pacientes son alérgicos, sería bastante más fácil extraer dicha información de la base de datos que revisar todos los cuestionarios de los pacientes.

El médico de este caso en particular, podría haber decidido omitir de la base de datos de pacientes los datos sobre alergia, y muchos de los otros campos del cuestionario. Seis semanas después, aparece en el mercado un nuevo medicamento que trata los síntomas de la fiebre del heno con efectos no deseables, de modo que el doctor desearía enviar una nota a todos los pacientes que tienen alergias al polen, con la descripción de este medicamento —situación que no se podía predecir.

Incluso cuando no se pueda predecir con seguridad todos los requerimientos futuros, hay que tratar de identificar *a priori* los campos que serán necesarios, y al mismo tiempo evaluar cuidadosamente cada campo para determinar si debe pertenecer a una de las bases de datos de dBASE IV.

El nuevo sistema de National Widgets

El nuevo sistema de base de datos para National Widgets debe asumir las funciones del presente sistema y reproducir todos los informes existentes. El sistema se instalará inicialmente en la computadora monousuario de la compañía, pero el propietario desea instalar una red de área local con tres estaciones de trabajo para final de año y al mismo tiempo añadir posibilidades multiusuario a la base de datos.

La conversión hacia operaciones mecanizadas ofrece una ventaja inmediata: elimina las entradas dobles. Por ejemplo, en el sistema manual, después de teclear las facturas, debería introducir la misma información en las fichas de los clientes y copiarla en el registro mensual de facturas; proceso tedioso, propenso a errores y que consume mucho tiempo.

Dado que el propietario y el archivador desean ejercer mayor control sobre las cuentas pendientes de cobro de la compañía, el nuevo sistema debe imprimir un informe de facturas por fecha ordenadas en orden alfabético por nombre de cliente. Para cada cliente, el informe debe listar todas las facturas abiertas fuera de balance

en las categorías de crédito estándar: de 0 a 30 días, 30 a 60 días, 60 a 90 días y más de 90 días. Esta información, también considerada de gran consumo de tiempo para calcularse a mano, deberá incluirse en el estado de cuentas mensual emitido por la computadora.

El propietario desea que el nuevo sistema le ayude a producir la información de contabilidad de forma más oportuna, así como a supervisar la línea de producción de la compañía, la política sobre existencias y la estructura de precios. Además del registro de facturas, la compañía necesita un listado de ingresos en efectivo y depósitos en bancos realizados en un determinado período de tiempo, generalmente diarios, semanales o mensuales. El archivador necesita un *resumen mensual* para listar todas las facturas, descuentos, gastos de envío, impuestos y pagos. Para ayudar a identificar las líneas de productos de mayor utilidad para la compañía y llevar a cabo una mejor planificación de la producción y los *stocks* de existencias, el sistema debe retener los pedidos al menos durante un año y generar un informe de resumen de pedidos que contabilice todos los productos solicitados en un inventario mensual.

Las funciones de correspondencia deben ampliarse para incluir la producción de cartas personalizadas en lugar de, o además de, las etiquetas adhesivas para los sobres. La compañía desea que el sistema sea capaz de identificar el equipo que posee el cliente y el tipo de productos que adquiere con mayor frecuencia. Manteniendo esta información, la producción de correspondencia selectiva se ejecutará de una forma más rápida e independiente del personal. Debería permitir también crear un perfil del cliente que incluya estadísticas sobre el número y el valor en efectivo de los pedidos de los clientes, agrupados bien por el equipo que poseen o por el tipo de productos que regularmente solicitan.

En el sistema manual, la única fuente que los empleados pueden consultar para saber si el cliente está ya en el archivo o para revisar su balance, es el conjunto de fichas que posee el encargado de archivar la información. Sería conveniente facilitar a los empleados que toman pedidos por teléfono un acceso más directo a los datos del cliente; aunque buscar un cliente en la pantalla puede que resulte tan complicado como sacarlo manualmente del paquete de fichas del fichero, en caso de que el sistema esté en uso o imprimiendo un informe extenso. El sistema debe, por tanto, imprimir una lista de referencia de clientes bien en papel continuo o en tarjetas Rolodex para que los empleados las guarden en sus escritorios. Por último, el nuevo programa debe archivar los clientes y borrar aquellos que no hayan realizado pedidos durante el año anterior (o en cualquier otro período que se elija).

Las nuevas funciones a ejecutar por el sistema de base de datos National Widgets se resumen en la Figura 1-5.

EXAMINANDO OTRO SOFTWARE

Suponiendo que existen paquetes comerciales para la aplicación que se está mecanizando, o para otra aplicación similar, tras un estudio de estos programas se puede llegar a la conclusión de "hacerlo uno mismo" con dBASE IV. Estos paquetes sugieren posibles campos adicionales, cálculos e informes. La misma importancia puede tener consultar otro software, como concentrar la atención en factores que produzcan pantallas de entrada de datos fáciles de usar, atractivas, y claras, así como informes legibles.

Para conseguirlo, las características de dBASE IV determinan la "apariencia y sensación" de los sistemas diseñados, y cuando se escriben programas para usuarios que intentan trabajar con dBASE IV fuera de su aplicación, es especialmente útil para conformar el estilo de las pantallas y menús de dBASE IV.

Datos adicionales del cliente:	Tipo de equipo que posee y productos pedidos.
Borrado de registro de cliente:	Retirar del fichero todos los clientes que no hayan realizado pedidos durante el último año.
Tarjetas índice o Rolodex:	Nombre del cliente, dirección, teléfono, contacto, fecha del primer pedido y tipo de equipo.
Cartas personalizadas:	Cartas impresas para todos los clientes, o selectivamente, según el equipo que poseen o los productos solicitados.
Perfil del cliente:	Resumen del número y el valor en efectivo de los pedidos realizados, según el equipo que poseen o los productos solicitados.
Informe de atrasos:	Resumen de las facturas pendientes de cobro de cada cliente, agrupadas por categorías estándar de retrasos y con subtotales.
Estado de cuentas mensuales:	Formato existente al que se la añade información relativa al retraso.
Listado de ingresos en efectivo:	Resumen de ingresos en efectivo durante un período de tiempo determinado.
Resumen mensual de envíos:	Resumen de totales de ventas, descuentos, impuestos, ventas en efectivo, cuentas pendientes y costo de las mercancías que utiliza el contable para preparar informes financieros.
Resumen de pedidos:	Resumen de pedidos por categoría de inventario, utilizado para el análisis de las ventas y la planificación de inventario.

Figura 1-5.

Requerimientos del sistema informatizado.

RESUMEN DE LAS OPERACIONES Y PLANIFICACION DEL TIEMPO

Una vez que ha realizado la descripción de los procedimientos manuales, que el nuevo sistema debe duplicar (así como añadir y mejorar las consideraciones estudiadas), está en condiciones de diseñar todos los componentes funcionales del nuevo sistema y establecer el programa de trabajo para la ejecución de cada operación. Si tiene conocimientos sobre programación y ha utilizado diagramas de flujo, es conveniente la realización de uno; pero si el análisis no necesita ser construido con tales estructuras formales, no es necesaria su realización. En lugar de, o además de, un diagrama de flujo puede realizarse un diseño del sistema mediante un listado de todas las entradas, el mantenimiento de archivos y las funciones de impresión o salidas que deberá producir el sistema.

Mediante el trabajo con las muestras de informes y las descripciones proporcionadas por el personal de la compañía, podrá ver con facilidad las funciones de la base de datos que no están duplicadas directamente en el sistema manual. Por ejemplo, la introducción de datos básicos del cliente y los procedimientos de actualización (introducción de nuevos clientes, búsqueda y modificación de datos, borrado de registros de clientes) tienen un procedimiento equivalente fácilmente identificable en el sistema manual: la actualización de las tarjetas de clientes. Sin embargo, la introducción de datos sobre pedidos puede amontonarse mientras se teclean las facturas, especialmente en la mente del empleado o del archivador, que procesan pedidos recibidos por correo; por otra parte, en el sistema informatizado, la introducción de datos de pedidos y la impresión de facturas son dos operaciones distintas.

El sistema de base de datos debe generar un conjunto de informes que no existen de la misma forma en el sistema manual. Para cada archivo de base de datos se debe introducir una lista de referencia (un listado completo) de los campos de todos los registros. La razón para la impresión de esta lista es doble. En primer lugar, con frecuencia demostrará ser la mejor forma de buscar información en la base de datos. Si el usuario quiere saber si un cliente está ya en el archivo de clientes, o quiere hacer una revisión rápida de un balance pendiente, es más fácil ver la lista de referencia de clientes que visualizar el registro en la pantalla, especialmente si alguien más está utilizando la computadora para otra tarea.

En segundo lugar, durante las etapas de prueba y depuración del desarrollo del sistema, las listas de referencia constituyen la mejor guía para saber exactamente qué información se tiene en la base de datos. Si, por ejemplo, se encuentra que ciertos productos no están contenidos en la factura, imprimiendo el archivo de la lista de referencia de pedidos puede determinarse si el problema está en la impresión de facturas o si el dato no está en los archivos, en cuyo caso la culpa puede ser del programa de introducción de pedidos, o bien, del operador.

El diseño final del sistema tiene varios propósitos. En primer lugar, ser una perspectiva más compacta y precisa del sistema; algo así como una lista de comprobación para asegurar que no se le ha olvidado nada cuando se escriben los programas. Acompañado de alguna descripción narrativa, puede ser la espina medular para la descripción de trabajo a ejecutar y, conforme se avance en el proyecto, se convierte en el núcleo de la documentación final del usuario.

La Figura 1-6 muestra el diseño final del sistema de cuentas pendientes de cobro de National Widgets.

A. Información del cliente:
   1. Actualizar información individual del cliente (añadir, consultar, cambiar, borrar).
      Analogía manual: Actualizar tarjetas de mayor de clientes.
      Cantidad: 550 clientes, 10-20 nuevos por mes.
      Frecuencia: Diaria.
   2. Archivar y borrar clientes inactivos.
      Analogía manual: Retirar tarjetas de clientes inactivos del archivo.
      Frecuencia: Una vez al año.
   3. Totales de clientes desde un año hasta la fecha.
      Analogía manual: Ninguna.
      Frecuencia: Una vez al año.
   4. Listas de clientes.
      Analogía manual: Tarjetas de mayor de clientes.
      a) Lista de referencia de clientes.
         Contenido: Todos los campos del cliente, todas las transacciones financieras.
         Frecuencia: Según demanda.
         Orden: Alfabético.
         Selecciones: Todos los clientes, nuevos clientes a partir de una fecha dada, o clientes sin actividad desde cierta fecha.
      b) Tarjetas índice o Rodolex.
         Contenido: Todos los campos del cliente.
         Frecuencia: Según demanda.
         Orden: Alfabético.
         Selecciones: Todos los clientes, uno, o nuevos clientes desde una fecha específica.
   5. Perfil de clientes.
      Analogía manual: Ninguna.
      Contenido: Desde hace un año hasta la fecha y el total en efectivo o número de pedidos.
      Frecuencia: Según demanda.
      Orden: Por tipo de equipo o categoría de productos.
      Selecciones: Por tipo de equipo o categoría de productos.
   6. Correspondencia a clientes.
      Analogía manual: Etiquetas o sobres escritos por empleados.
      a) Impresión de etiquetas para todos los empleados.
         Cantidad: 550 clientes (1.500 anticipadas).
         Frecuencia: Dos veces al año.
         Orden: Por código postal.
      b) Impresión de etiquetas para clientes seleccionados.
         Cantidad: 50-250 por ejecución.
         Frecuencia: 5-6 veces al año.
         Orden: Por código postal.
         Selecciones: Por tipo de producto, tipo de equipo, fecha del último pedido, total de pedidos.

Figura 1-6.

Diseño del sistema de cuentas pendientes de National Widgets.

c) Impresión de cartas para clientes seleccionados.
   Cantidad: 50-250 por ejecución.
   Frecuencia: 5-6 veces al año.
   Orden: Por código postal.
   Selecciones: Por tipo de producto, tipo de equipo, fecha del último pedido, total de pedidos.

d) Impresión de sobres para tarjetas de Navidad.
   Cantidad: 550 clientes.
   Frecuencia: Una vez al año.
   Orden: Alfabético.

B. Procesos de pedidos:

1. Actualización de pedidos (añadir, consultar, cambiar, borrar).
   Analogía manual: Formas de pedidos.
   Cantidad: 1.500 pedidos al mes.
   Frecuencia: Diaria.
2. Impresión de la Lista de Referencia de pedidos:
   Analogía manual: Ninguna.
   Contenido: Todos los campos del pedido.
   Frecuencia: Según demanda.
   Orden: Alfabético por cliente y número de factura, o por número de cliente.
   Selecciones: Todos los pedidos actuales, todos los pedidos del archivo histórico, o un cliente a la vez.
3. Impresión de pedidos para el almacén.
   Analogía manual: Copia de factura emitida.
   Cantidad: 1.500 pedidos al mes.
   Frecuencia: Diaria.
4. Impresión de facturas.
   Analogía manual: Factura tecleada.
   Cantidad: 1.500 pedidos al mes.
   Frecuencia: 2-3 veces a la semana.
5. Información de facturas enviadas al archivo de clientes.
   Analogía manual: Actualización de tarjetas de clientes.
   Frecuencia: Después de imprimir cada lote de facturas.
6. Actualización del archivo histórico de pedidos.
   Analogía manual: Facturas archivadas en orden cronológico.
   Frecuencia: Después de imprimir cada lote de facturas.
7. Depuración y reinicialización del archivo histórico.
   Analogía manual: Llevar las facturas antiguas a su lugar de almacenamiento.
   Frecuencia: Anual.

C. Transacciones financieras:

1. Actualización de pedidos (añadir, consultar, cambiar, borrar).
   Analogía manual: Actualización de tarjetas de clientes.
   Cantidad: 1.500 pedidos y 1.500 pagos mensuales.
   Frecuencia: Diaria.
2. Impresión de lista de referencia de transacciones.
   Analogía manual: Ninguna.

Figura 1-6.

Diseño del sistema de cuentas pendientes de National Widgets *(continuación)*.

   Contenido: Todos los campos de pagos y facturas.
   Frecuencia: Mensual o según demanda.
   Orden: Alfabético por cliente y número de factura.
   Selecciones: Todas las transacciones actuales, todas las transacciones del archivo histórico o un cliente a la vez.
3. Impresión de registro de facturas.
   Analogía manual: Hoja de cálculo Lotus 1-2-3.
   Contenido: Todos los campos de facturas.
   Frecuencia: Mensual.
   Orden: Por número de factura.
   Selecciones: Rango especificado de número de factura o fechas.
4. Listado de ingresos en efectivo.
   Analogía manual: Depósitos bancarios.
   Contenido: Todos los campos de pago.
   Orden: Cronológicamente por fecha de pago.
   Selecciones: Rango especificado de fechas de pago.
5. Impresión de informes de retraso.
   Analogía manual: Ninguna.
   Contenido: Todos los campos de pagos y facturas.
   Frecuencia: Mensual.
   Orden: Cronológicamente por fecha de pago.
   Selecciones: Todas las facturas no pagadas.
6. Impresión de estados de cuentas mensuales.
   Analogía manual: Recordatorios tecleados.
   Contenido: Nombre y dirección del cliente, todas las facturas y pagos del mes.
   Frecuencia: Mensual.
   Orden: Alfabético.
   Selecciones: Todos los clientes con balance atrasado.
7. Resumen de pedidos.
   Analogía manual: Ninguna.
   Contenido: Todos los campos de pedidos.
   Frecuencia: 2-3 veces al año.
   Orden: Por categoría de inventario y por número de referencia.
   Selecciones: Todos los productos, por categoría o tipo de equipo.
8. Resúmenes de contabilidad.
   Analogía manual: Resumen de contabilidad anual.
   Contenido: Total de ventas, descuentos, impuestos, ingresos en efectivo, cuentas pendientes y costos de las mercancías.
   Frecuencia: Mensual.
   Selecciones: Transacciones mensuales o anuales.
9. Actualización del archivo histórico de transacciones.
   Analogía manual: Ninguna.
   Frecuencia: Mensual.
10. Depuración y reinicialización del archivo histórico de transacciones.
    Analogía manual: Ninguna.
    Frecuencia: Según demanda.

Figura 1-6.

Diseño del sistema de cuentas pendientes de National Widgets *(continuación)*.

CAPITULO VEINTISEIS

Utilidades de mantenimiento de una base de datos

Además de los programas que son necesarios para llevar a cabo la entrada de datos, consultas, proceso, y funciones de salida solicitadas por los usuarios, todas las aplicaciones de dBASE IV requieren *programas de utilidad*: programas que llevan a cabo las actividades de mantenimiento de los archivos en disco y las bases de datos necesarias para mantener el sistema sin la asistencia de un usuario entendido técnicamente. Con frecuencia, estas operaciones no están incluidas en la descripción original del sistema proporcionada por los usuarios, ya que no siempre realizan operaciones manuales análogas y llevan a cabo el mantenimiento manual del sistema de una forma más fácil o menos esencial. Por ejemplo, National Widgets puede que no se deshaga de los archivos de clientes que no han realizado un pedido en varios años; pero si no proporciona una opción de depuración para ese fin en la aplicación informatizada, puede que se le acabe el espacio del disco después de un año o dos.

Este capítulo describe algunos programas de utilidad típicos que llevan a cabo actividades de mantenimiento de una base de datos y procedimientos de recuperación de fallos tales como reindexación, reconstrucción de campos de totales basados en los datos de la transacción, o almacenamiento y depuración de registros obsoletos para compactar las bases de datos, conservar el espacio de disco, y mejorar el rendimiento. Los programas de utilidad también permiten que los usuarios lleven a cabo operaciones que en cualuier otro caso requerían técnicas más refinadas que las que probablemente poseen, incluso si están familiarizados con dBASE IV, tal como copiar datos en los formatos de archivo externo requeridos para intercambiar datos con otras computadoras y paquetes de software. El Capítulo 27 describe otra clase de programas de utilidad que pueden ser descritos como utilidades de *mantenimiento del*

sistema, ya que dependen menos de la naturaleza de los datos almacenados en los archivos que manipulan.

Siempre que sea posible hacer eso, debería procurar escribir programas de utilidad que sean bastante generales para utilizarlos en más de una aplicación. Aunque no puede escribir un programa de depuración y almacenamiento de archivos que trabaje con cualquier combinación de archivos de bases de datos relacionados, utilidades tales como un programa para empaquetar y reindexar las bases de datos del sistema pueden ser independientes de la aplicación. Este capítulo describe algunas técnicas para escribir programas de utilidad que funcionen en todas o en la mayoría de sus aplicaciones de dBASE IV. Los programas presentados en este capítulo y en el siguiente se pueden utilizar, modificándolos si es necesario, como el núcleo de su biblioteca de utilidades personales para satisfacer sus preferencias personales o las necesidades de sus usuarios, así como los procedimientos generales descritos en los cinco últimos capítulos formaron su biblioteca básica de procedimientos.

USO DE CATALOGOS PARA SOPORTAR LAS UTILIDADES DE PROPOSITO GENERAL

El Capítulo 3 describió cómo utilizar el catálogo de dBASE IV, que almacena una lista de los archivos en disco utilizados en una aplicación, para ayudarle a documentar el sistema y monitorizar el proceso de desarrollo. También puede poner el catálogo en uso para escribir programas de utilidad de propósito general que sean independientes del nombre de los archivos sobre los que operan. Por ejemplo, el catálogo que describe las bases de datos podría suministrar su contenido en una lista de archivos presentada a los usuarios para que estos seleccionen los archivos para hacer copias de seguridad, restaurar o copiar en un formato externo, así como un programa que empaquete y reindexe todos los archivos de una aplicación podría leer el nombre de las bases de datos de un catálogo. Excluyendo todas las referencias explícitas a nombres de archivo en sus programas de utilidad, puede estar seguro de que ellos continuarán funcionando correctamente si añade nuevos archivos a la aplicación o borra los archivos que no necesita. Además puede aumentar la productividad de la programación: con frecuencia puede utilizar el mismo programa de utilidades en todas las aplicaciones que escriba.

PROGRAMAS PARA EMPAQUETAR Y REINDEXAR LAS BASES DE DATOS

El Capítulo 9 introdujo la primera versión de un programa que reindexaba todas las bases de datos del sistema National Widgets para permitir que los usuarios recuperen más fácilmente un fallo del software o del disco. Pues bien, podría fácilmente añadir a este programa un paso opcional que empaquete las bases de datos para eliminar los registros marcados antes de reindexarlos, de modo que el nuevo programa sirva como una rutina de mantenimiento de bases de datos y para recuperar fallos. Puesto que dBASE IV reconstruye automáticamente todas las etiquetas contenidas en un archivo de producción .MDX después de ejecutar una orden PACK, si el usuario elige empaquetar los archivos, el programa de reindexación no necesita saber el nombre de las etiquetas o las expresiones clave del índice. Si está dispuesto a utilizar la orden REINDEX (en lugar de reconstruir los índices desde el editor con INDEX ON), el paso de la reindexación también puede ser independiente del nombre de los archivos, nombre de las etiquetas y expresiones clave de la aplicación, y el programa puede leer el nombre de las bases de datos a partir del catálogo.

En la Figura 26-1 se lista un programa de reindexación llamado NWINDEX2.PRG que ilustra esta estrategia. Este programa supone que el archivo de catálogo del sistema ha sido abierto en el programa del menú de utilidades con la siguiente orden:

```
USE NW.CAT ALIAS Filelist
```

Cuando abre un catálogo con una orden USE, dBASE IV trata el catálogo como cualquier otra base de datos, incluso si la abre en el área de trabajo 10, que normalmente se utiliza para un catálogo abierto con SET CATALOG. Abrir el catálogo con USE no hace que el proceso de actualización de catálogo se active automáticamente cuando utiliza SET CATALOG. Además le permite asignar a la base de datos de catálogo un alias, un índice y establecer una condición de filtro para procesar los registros de catálogo de forma selectiva, en otras palabras, tratar el catálogo exactamente como un archivo de base de datos ordinario que contiene una lista de los archivos en disco utilizados en su aplicación. Colocando la orden USE en el programa que llama, asegura que NWINDEX2.PRG no contiene nombres de archivos específicos de la aplicación y funciona en cualquier sistema de base de datos.

El programa NWINDEX2.PRG pide al usuario si desea empaquetar o reindexar y almacena las respuestas en las variables de memoria MPACK y MINDEX. Si cualquiera de las variables de memoria en .T., es decir, el usuario elige realizar al menos una de las dos operaciones, el programa selecciona el área de trabajo FILELIST y establece una condición de filtro basada en el valor del campo TYPE para seleccionar los registros del catálogo que representan archivos de bases de datos (recuerde del Capítulo 3 que el campo TYPE almacena las extensiones estándar del tipo de archivo representado por la entrada del catálogo, independientemente de la extensión actual asignada al archivo).

El programa utiliza un bucle SCAN para examinar las entradas de la base de datos en el catálogo. Puesto que la sentencia SCAN sin una cláusula FOR o WHILE hace que dBASE IV se desplace al principio de la base de datos seleccionada antes de empezar a leer los registros, no tiene que volver a posicionar el puntero de registro con la orden GOTO TOP después de la orden SET FILTER. Con el bucle SCAN, el programa selecciona una segunda área de trabajo y abre el archivo descrito por el registro actual del catálogo. Note el uso del paréntesis para hacer que dBASE IV lea la expresión FILELIST->PATH como un nombre de archivo, esta *sustitución del nombre del archivo* es similar a la sustitución macro efectuada por el símbolo &. Para permitir que el usuario monitorice su proceso, el programa de reindexación visualiza un mensaje de estado que incluye el nombre del archivo que está en proceso con la función DBF, que da por resultado el nombre de la base de datos abierta en el área de trabajo seleccionada.

```
* NWINDEX2.PRG
* PROGRAMA PARA EMPAQUETAR Y/O REINDEXAR TODAS LAS BASES DE DATOS
* ULTIMA REVISION: 08/06/89 M.LISKIN

DO Scrnhead WITH "", "Empaquetar y reindexar todas las bases de datos"

DEFINE WINDOW utility FROM 5,10 TO 20,70
ACTIVATE WINDOW utility

mpack = .T.                              && ¿Empaquetar las bases de da-
tos?
mindex = .T.                             && ¿Reindexar las bases de datos?
@ 1, 5 SAY "¿Desea eliminar los registros marcados? (S/N)" GET mpack;
       PICTURE "Y"
@ 3, 5 SAY "¿Desea reindexar todas las bases de datos? (S/N)" GET mindex;
       PICTURE "Y"
READ

IF mpack .OR. mindex

   SELECT Filelist                       && Area de trabajo del catálogo
   SET FILTER TO UPPER(Type) = "DBF"     && Selecciona las entradas de la
                                            base datos

   SCAN
      SELECT 2                           && Abre las bases de datos des-
                                            critas
      USE (Filelist->Path)               && por el registro del catálogo
                                            actual
      @ 6, 0 CLEAR
      DO Center WITH 6,60, "", "Operando con " + DBF()
      SET TALK ON
      IF mpack                           && Si el usuario elige empaque-
                                            tar,
         PACK                            &&  empaqueta y reindexa
      ELSE
         REINDEX                         && Sólo reindexa
      ENDIF
      SET TALK OFF
      SELECT Filelist
   ENDSCAN

ENDIF
```

Figura 26-1.

Un programa de uso general para empaquetar y reindexar todas las bases de datos.


```
CLOSE DATABASES
DEACTIVATE WINDOW utility
RELEASE WINDOW utility
RETURN

* Fin del programa NWINDEX2.PRG
```

Figura 26-1.

Un programa de uso general para empaquetar y reindexar todas las bases de datos *(continuación)*.

Para cada base de datos procesada en el bucle SCAN, el programa de reindexación comprueba el valor de MPACK para determinar, si debe empaquetar el archivo. Si el programa no ejecuta la orden PACK, no es necesario ejecutar la orden REINDEX, ya que dBASE IV reconstruye automáticamente todas las etiquetas del archivo de producción .MDX después de empaquetar una base de datos. Si MPACK es .F., el usuario ha elegido reindexar sin eliminar los registros marcados, el programa ejecuta una orden REINDEX en lugar de PACK. En cualquier caso, el programa establece el parámetro TALK en ON para permitir que el usuario monitorice el progreso de este proceso y lo devuelve a OFF cuando termina la operación.

Observe que puesto que la orden PACK reindexa automáticamente sólo necesita hacer una pregunta al principio del programa NWINDEX2.PRG. Si el usuario elige empaquetar las bases de datos respondiendo "Y" a la pregunta inicial ("¿Desea eliminar los registros marcados?") la segunda pregunta ("¿Desea reindexar todas las bases de datos?") es redundente. La variable de memoria utilizada para captar la respuesta a la segunda pregunta se utiliza solamente si la estructura IF comprueba que el usuario ha respondido "Y" al menos a una de las preguntas. La segunta pregunta "¿Desea reindexar todas las bases de datos?", se incluye principalmente para expresar la pregunta en términos que no requieren ninguna familiaridad con las operaciones de las órdenes PACK y REINDEX, y para enfatizar el hecho de que existen dos elecciones independientes. Si el usuario responde "N" a ambas preguntas, el programa no hace nada.

El uso de un catálogo para identificar las bases de datos en una aplicación da por resultado un programa de reindexación y empaquetamiento conciso y muy general, pero en la mayoría de los casos esto no es la mejor aproximación. Los usuarios puede que deseen ejecutar un programa de reindexación bajo circunstancias en las que uno o más archivos de producción .MDX son ilegibles o no están disponibles, tal como después de formatear o sustituir el disco y luego recuperar las bases de datos desde copias de seguridad en discos flexibles o después de que un disco haya dañado físicamente uno de los archivos .MDX y dBASE IV no pueda leer las expresiones clave y el nombre de las etiquetas.

Una alternativa es mantener la estructura del programa utilizada en NWINDEX.PRG (el programa descrito en el Capítulo 9) y añadir las elecciones condicionales (empaquetar o reindexar) junto con unas mejoras en la visualización del estado. La Figura 26-2 lista NWINDEX3.PRG, un programa de este tipo para el sistema Na-

```
* NWINDEX3.PRG
* PROGRAMA PARA RECONSTRUIR TODOS LOS INDICES DE NATIONAL WIDGETS
* ULTIMA REVISION 08/06/89   M.LISKIN

DO Scrnhead WITH "", "Empaquetar y reindexar todas las bases de datos"

DEFINE WINDOW utility FROM 5,10 TO 20,70
ACTIVATE WINDIDOW utility

mpack = .T.
mindex = .T.
@ 1, 5 SAY "¿Desea eliminar los registros marcados? (Y/N)" GET mpack;
       PICTURE "Y"
@ 3, 5 SAY "¿Desea reindexar todas las bases de datos? (Y/N)" GET mindex;
       PICTURE "Y"
READ

IF mpack .OR. mindex

PLAY MACRO P
DELETE FILE NWcust .mdx
USE NWcust
CLEAR
DO Center WITH 6, 60, "", "Trabajando con el archivo Clientes"
SET TALK ON
IF mpack
   PACK
ENDIF
INDEX ON Account TAG Account
INDEX ON Zip TAG Zip
SET TALK OFF

DELETE FILE NWinvent.mdx
USE NWinvent
CLEAR
DO Center WITH 6, 60, "", "Trabajando con el archivo Inventarios"
SET TALK ON
IF mpack
   PACK
ENDIF
INDEX ON Partnumber TAG Partnumber
INDEX ON Category + Partnumber TAG Catpart
SET TALK OFF

DELETE FILE NWorder.mdx
USE NWorder
CLEAR
```

Figura 26-2.

Un programa para empaquetar y reindexar las bases de datos de National Widgets.


```
DO Center WITH 6, 60, "", "Trabajando con el archivo Pedidos"
SET TALK ON
IF mpack
   PACK
ENDIF
INDEX ON Account + Invoice TAG Account
INDEX ON Invoice TAG Invoice
SET TALK OFF

DELETE FILE NWohist.mdx
USE NWohist
CLEAR
DO Center WITH 6, 60, "", "Trabjando con el archivo Histórico de Pedidos"
SET TALK ON
IF mpack
   PACK

ENDIF
INDEX ON Account + Invoice TAG Account
INDEX ON Invoice TAG Invoice
SET TALK OFF

DELETE FILE NWtxn.mdx
USE NWtxn
CLEAR
DO Center WITH 6, 60, "", "Trabajando con el archivo Transacción"
SET TALK ON
IF mpack
   PACK
ENDIF
INDEX ON Account + Invoice TAG Account
INDEX ON Invoice TAG Invoice
SET TALK OFF

DELETE FILE NWthist.mdx
USE NWthist
CLEAR
DO Center WITH 6, 60, "", "Trabajando con el archivo Histórico de Transac-
ciones"
SET TALK ON
IF mpack
   PACK
ENDIF
INDEX ON Codetype + Invoice TAG Account
INDEX ON Invoice TAG Invoice
SET TALK OFF
```

Figura 26-2.

Un programa para empaquetar y reindexar las bases de datos de National Widgets *(continuación)*.

```
   DELETE FILE NWcodes.mdx
   USE NWcodes
   CLEAR
   DO Center WITH 6, 60; "", "Trabajando con el archivo Códigos"
   SET TALK ON
   IF mpack
      PACK
   ENDIF
   INDEX ON Codetype + Code TAG Codetype
   SET TALK OFF

ENDIF

USE
DEACTIVATE WINDOW utility
RELEASE WINDOW utility
SET SAFETY ON

* Fin del programa NWINDEX3.PRG
```

Figura 26-2.

Un programa para empaquetar y reindexar las bases de datos de National Widgets *(continuación)*.

tional Widgets. Puesto que este programa incluye el archivo Códigos así como las seis bases de datos originales, debe modificar la macro P descrita en el Capítulo 9 de modo que conste de siete caracteres "P". Aunque esta aproximación resuelve el problema de determinar las expresiones clave de índices, el hecho de que las expresiones clave estén "codificadas permanentemente" significa que debe escribir una versión diferente del programa de reindexación para cada aplicación, y debe modificar el programa cada vez que añade una nueva base de datos al sistema y cada vez que añade o modifica una expresión clave de índice.

Otro método es crear una base de datos que sirva como tipo del archivo de catálogo mejorado. Además de los registros que describen las bases de datos en la aplicación, puede añadir un registro para cada etiqueta de índices que registre la expresión clave. La Figura 26-3 lista la estructura de este archivo, NWFILES.DBF, y su contenido para el sistema National Widgets; una versión del programa de reindexación que lee este archivo en lugar del catálogo aparece en la Figura 26-4 (el campo TAG, que es análogo al campo con el mismo nombre, en el catálogo se utiliza en la versión multiusuario del programa de reindexación descrita en el Capítulo 29).

Nótese que en lugar de crear un nuevo archivo, podría añadir registros a un catálogo de archivo de dBASE IV para representar las etiquetas de índice, colocar el nombre de las etiquetas en el campo PATH o FILE_NAME, la expresión clave en el campo TITLE (similar al modo en que dBASE IV trata las entradas del catálogo para los índices .NDX), y un identificador tal como "MDX" o "TAG" en el campo TYPE. No obstante, no debería permitir que dBASE IV actualice el catálogo después de añadir estos registros.


```
Estructura de la base de datos: C:\DBASE4\NWFILES.DBF
Número de registros          :     13
Fecha de última actualización: 08/08/89
Campo  Nombre      Tipo        Ancho   Dec     Indi
    1  DATABASE    Carácter       50           Y
    2  TAGNAME     Carácter       10           N
    3  INDEXKEY    Carácter      100           N
    4  TAG         Carácter        4           N
** Total **                      165

Record#  DATABASE                  TAGNAME     INDEXKEY                TAG
      1  NWCUST                    ACCOUNT     Account
      2  NWCUST                    ZIP         Zip
      3  NWINVENT                  PARTNUMBER  Partnumber
      4  NWINVENT                  CATPART     Category + Partnumber
      5  NWORDER                   ACCOUNT     Account + Invoice
      6  NWORDER                   INVOICE     Invoice
      7  NWTXN                     INVOICE     Invoice
      8  NWTXN                     ACCOUNT     Account + Invoice
      9  NWOHIST                   ACCOUNT     Account + Invoice
     10  NWOHIST                   INVOICE     Invoice
     11  NWTHIST                   INVOICE     Invoice
     12  NWTHIST                   ACCOUNT     Account + Invoice
     13  NWCODES                   CODETYPE    Codetype + Code
```

Figura 26-3.

El archivo Archivos de National Widgets.

El programa NWINDEX4.PRG, al igual que las versiones anteriores presentadas en este capítulo, suponen que el archivo Archivos ha sido abierto en el programa del menú de utilidades con la orden USE que le asignó el alias FILELIST. El programa de reindexación también utiliza un bucle SCAN para examinar el archivo Archivos, y la secuencia del proceso depende del hecho de que las bases de datos que almacenan la lista de archivos están indexadas por DATABASE. Este reconstruye esta etiqueta de forma incondicional para asegurar que el índice se completa y se actualiza.

Cada registro del archivo Archivos representa una etiqueta de índices, y el programa debería reconstruir un índice en cada paso por el bucle SCAN. Sin embargo, cada base de datos representada en el archivo Archivos debería ser empaquetada una sola vez. El programa utiliza una variable de memoria llamada MDATABASE para retener el nombre de la base de datos procesada en la última pasada a través del bucle SCAN y ejecuta la orden PACK solamente cuando el nombre de la base de datos del registro actual del archivo Archivos es diferente de la última. MDATABASE se inicializa como un espacio en blanco antes de la sentencia SCAN de modo que la primera vez que se pasa a través del bucle sea diferente al campo DATABASE y el programa empaquete la primera base de datos.

```
* NWINDEX4.PRG
* PROGRAMA PARA EMPAQUETAR Y/O REINDEXAR TODAS LAS BASES DE DATOS
* ULTIMA REVISION:   08/06/89 M.LISKIN

DO Scrnhead WITH "", "Empaquetar y reindexar todas las bases de datos"

DEFINE WINDOW utility FROM 5,10 TO 20,70
ACTIVATE WINDOW utility

mpack = .T.                                  && ¿Empaquetar las bases de da-
                                                tos?
mindex = .T.                                 && ¿Reindexar las bases de datos?
@ 1, 5 SAY "¿Desea eliminar los registros marcados? (S/N)" GET mpack;
       PICTURE "Y"
@ 3, 5 SAY "¿Desea reindexar tods las bases de datos? (S/N)" GET mindex;
       PICTURE "Y"
READ

IF mpack .OR. mindex

   SELECT Filelist                           && Area de trabajo del archivo
                                                Archivos
   INDEX ON Database TAG Database
   GOTO TOP
   mdatabase = ""              && Ultima base de datos procesada

   SCAN
      mtagname = TRIM(Tagname)               && Nombre de la etiqueta
      mindexkey = TRIM(Indexkey)             && Expresión clave del índice

      IF Database <> mdatabase
         mdatabase = TRIM(Database)          && Ultima base de datos procesada
         mindexfile = TRIM(Database) + ".MDX"     && Nombre del archivo MDX
         DELETE FILE (mindexfile)            && Borra el archivo MDX
         SELECT 2
         USE (mdatabase)                     && Abre el archivo de base de da-
                                                tos
         CLEAR
         DO Center WITH 6, 60, "", "Operando con " + DBF()
         IF mpack                            && Si el usuario elige empaquetar,

            SET TALK ON
            PACK                             && Sólo empaqueta
            SET TALK OFF
         ENDIF
      ENDIF
```

Figura 26-4.

Un programa para reindexar y empaquetar que consulta el archivo Archivos.


```
      SELECT 2
      SET TALK ON
      INDEX ON &mindexkey. TAG &mtagname.    && Reconstruye la etiqueta de ín-
                                                dices
      SET TALK OFF
      SELECT Filelist

   ENDSCAN

ENDIF

CLOSE DATABASES
DEACTIVE WINDOW utility

RELEASE WINDOW utility
RETURN

* Fin del programa NWINDEX4.PRG
```

Figura 26-4.

Un programa para reindexar y empaquetar que consulta el archivo Archivos *(continuación)*.

En cada paso por el bucle SCAN, el programa NWINDEX4.PRG recorta el nombre de la etiqueta y la clave de índice y almacena el resultado en las variables de memoria MTAGNAME y MINDEXKEY. Si el contenido del campo DATABASE es diferente al de MDATABASE, es decir, el registro actual del archivo Archivos es el primero para una base de datos dada, el programa establece MDATABASE en cero para que coincida con el campo DATABASE, almacena el nombre del archivo de producción .MDX en la variable de memoria MINDEXFILE (el mismo nombre que la base de datos con la extensión .MDX) y utiliza esta variable para identificar el archivo de producción .MDX en una orden DELETE FILE.

El nuevo programa de reindexación no puede utilizar la macro P, definida en el Capítulo 9, para responder al mensaje de error que se visualiza cuando dBASE IV intenta abrir una base de datos para la cual el archivo de producción .MDX es erróneo. La orden PLAY MACRO debe ser repetida una vez para cada base de datos procesada, pero no puede utilizar una única macro que contenga varios caracteres "P" para responder a los diálogos de corrección de errores sucesivos, ya que el número de veces que dBASE IV visualiza el mensaje de error depende del número de base de datos representado en el archivo Archivos. Cabe pensar en crear una macro que contenga solamente una "P" e incluir la orden PLAY MACRO que ejecuta esta macro en el bucle SCAN, pero esta estrategia está descartada por el hecho de que llamadas repetidas a la misma macro mantienen el mensaje de error. En su lugar, el programa NWINDEX4.PRG permite que el usuario responda a la caja de diálogo visualizada si el archivo de producción .MDX es incorrecto. Si adopta esta estrategia, puede hacer una copia de seguridad de los índices así como de las bases de datos.

Formato archivo	Palabra de TYPE	Extensión	Ordenes Importar/ Exportar
Delimited	DELIMITED	.TXT	COPY, APPEND
Longitud fija	SDF	.TXT	COPY, APPEND
Lotus 1-2-3 versión 1A	WKS	.WKS	COPY, APPEND
Lotus 1-2-3 versión 2	WK1	.WK1	IMPORT
Framework II	FW2	.FW2	COPY, EXPORT, APPEND IMPORT
RapidFile	RPD	.RPD	COPY, EXPORT, APPEND, IMPORT
dBASE II	DBASEII	.DB2	COPY
dBASE III/III PLUS	DBMEMO3	.DBF	COPY
PFS:FILE	PFS	none	EXPORT, IMPORT
VisiCalc	DIF	.DIF	COPY, APPEND
Multiplan	SYLK	none	COPY, APPEND

Tabla 26-1.

Formatos de intercambio de datos de dBASE IV.

formatos de archivo es que no está disponible. Por ejemplo, debido a que dBASE IV puede leer una base de datos creada por dBASE III o dBASE III PLUS directamente, no hay necesidad de una orden especial para abrir los registros de un archivo de base de datos de dBASE III o dBASE III PLUS. No obstante, dBASE IV almacena los campos memo de forma diferente y dBASE III y dBASE III PLUS no pueden leer las bases de datos de dBASE IV que contienen campos memo. En su lugar, debe copiar el archivo en un formato de dBASE III utilizando una orden COPY que incluya la palabra clave DBMEMO3. Similarmente, puesto que la versión 2 de Lotus 1-2-3 puede leer un archivo .WKS (en el formato de hoja electrónica de la versión 1A) no es necesario exportar archivos a un formato WK1 de Lotus.

Puede ejecutar la orden IMPORT sin abrir primero la base de datos, ya que todos los formatos de archivo soportados por IMPORT incluyen la información de la estructura requerida para construir el correspondiente archivo de base de datos de dBASE IV desde el principio. Cuando utiliza la orden IMPORT, dBASE IV convierte todo el archivo externo; más tarde, debe eliminar cualquier registro o campo innecesario. En contraste, EXPORT, APPEND y COPY requieren que la base de datos esté abierta. Puede utilizar estas órdenes para leer o escribir archivos que contengan sólo datos con una información no estructurada. Además, el tipo de archivo que desea leer o escribir se identifica añadiendo una cláusula TYPE a la orden APPEND, COPY, IMPORT o EXPORT. La sintaxis es la siguiente:

COPY TO *nombre de archivo* TYPE *tipo de archivo*

o

EXPORT TO *nombre de archivo* TYPE *tipo de archivo*

o

APPEND FROM *nombre de archivo* TYPE *tipo de archivo*

o

IMPORT FROM *nombre de archivo* TYPE *tipo de archivo*

Cuando utiliza la orden COPY o EXPORT para escribir los datos en un formato externo, puede determinar el orden en que se copian los registros ordenando e indexando el archivo origen y puede utilizar cualquier combinación de filtros, alcance, cláusula FOR, y cláusula WHILE. Además, puede añadir una cláusula FIELDS a una orden COPY o EXPORT para transferir los campos seleccionados, en el orden en que ellos aparecen en la lista de campos, al archivo externo. No puede copiar campos memo y cuando implica a la palabra clave FIELDS sólo puede copiar los campos, *no los resultados de la evaluación de las expresiones* basadas en los campos de la base de datos. En este capítulo se describen algunos métodos para superar estas limitaciones.

Consideraciones especiales para los archivos de texto

Dos de los formatos de archivo estándar, el delimitado y el SDF, pueden ser utilizados para intercambiar datos con una gran variedad de programas, incluyendo aquéllos que se ejecutan bajo sistemas operativos diferentes o en minicomputadoras. En algunos casos, puede utilizar alguno de estos formatos como medio de transferencia entre dos programas cuando no puede leer directamente un formato que puede producir el otro. Debido a su amplitud de uso, estos formatos merecen una atención especial, incluso si su uso no siempre representa el modo más directo de intercambiar datos con otro software.

Un archivo *delimitado* es un archivo de texto ASCII en el que los campos están separados por comas y los campos de caracteres están encerrados (limitados) por signos de puntuación. Aunque el delimitador estándar son las comillas ("), puede especificar otro diferente. Puesto que cada registro finaliza con un retorno de carro, cuando utilice una orden TYPE de DOS o de dBASE IV para visualizar el contenido de un archivo delimitado, cada registro aparecerá en una línea de la pantalla.

Puede leer un archivo delimitado en una hoja de trabajo de Lotus 1-2-3 existente utilizando la secuencia de órdenes /File Import Numbers. Muchos programas de tratamiento de textos que tienen posibilidades de escritos personalizados, tales como WordStar, Microsoft Word, y Volkswriter, pueden leer este formato directamente y combinar los campos de la base de datos con un texto fijo para generar formularios tal como cartas personalizadas. Otros programas, incluyendo WordPerfect, que almacenan archivos de personalización en un formato propio y no pueden leer archivos delimitados directamente, proporcionan un programa de utilidad que convierte un archivo delimitado en el formato de almacenamiento interno requerido.

La Figura 26-5 ilustra el archivo delimitado producido por la siguiente secuencia

```
"Gina Aronoff", "", "601 First Street", "", "Benicia", "CA", "94510", 19881214
"Barbara Goddard", "Ellis Manufacturing", "6387 130th N.E.", "", "Bellevue", "WA", "98005", 1988021
"Lisa Burns", "The Image Makers", "1900 Powell Street", "", "EWmeryville", "CA", "94608", 19871112
"Carolyn Sumner", "J. Thomas Johnson, CPA", "50 California Street", "San Francisco", "CA", "9411", 19881206
"Richard Kelly", "Kelly and Sons Furniture", "14800 Bancroft Avenue", "", "San Leandro", "CA", "04578", 19890123
"Judy Barnes", "Carol Klein, M.D.", "1849 S.E. 40th Avenue", "", "Portland", "OR", "97214", 1988012.
"Joan Mills", "Lewis and Associates", "460 Grand Avenue", "", "Oakland", "CA", "94610", 198980129
"Kathy McDonald", "RapidType Secretarial Svc", "2457 Union Street", "", "San Francisco", "CA", "94123", 19890310
```

Figura 26-5.

Un archivo delimitado.

de órdenes, las cuales copian los campos nombre, dirección y fecha del último pedido del archivo de Clientes de National Widgets para los clientes que tienen un equipo IBM:

```
USE NWcust ORDER Account
COPY TO NWIBM1 TYPE DELIMITED FIELDS Contact, Company, Address1,;
   Address2, City, State, Zip, Lastorder FOR "IBM" $ Equipment
```

El archivo delimitado no conoce el nombre de los campos o cualquier otra descripción explícita de la estructura del archivo. Los programas que leen y escriben archivos delimitados identifican los campos contando las comas desde el principio del registro: el Campo 1 consta de todos los caracteres hasta la primera coma, el Campo 2 todos los caracteres entre la primera y la segunda coma, y así sucesivamente. Las comillas que rodean a los campos de caracteres son necesarias para distinguir las comas incluidas (como en "J. Thomas Johnson, CPA") de aquellas utilizadas como separadores de campo. De forma similar, si un campo está vacío, la coma que normalmente seguirá a este campo permanece como un lugar vacío para que no se confunda con el siguiente campo.

Si sus datos incluyen comillas, o si el programa con el que necesita intercambiar datos requiere un delimitador diferente, puede anular las comillas por omisión especificando su propio delimitador de caracteres. Por ejemplo, podría copiar datos desde dBASE IV en un archivo delimitado llamado CUSTOMER.TXT, en el que los campos de caracteres están rodeados por barras inclinadas, con la siguiente orden:

```
COPY TO CUSTOMER TYPE DELIMITED WITH /
```

Independientemente del carácter delimitador, dBASE IV siempre separa los campos con comas. Puede eliminar ambos delimitadores y los separadores de campo (las comas), y en lugar de insertar un espacio en blanco único entre los campos, sustituir la palabra clave BLANK por el carácter delimitador del siguiente modo:

```
COPY TO NWLETTER TYPE DELIMITED WITH BLANK
```

Este formato es más útil para copiar datos numéricos, y muchos datos estadísticos y programas de análisis numérico, incluyendo aquellos que se ejecutan en minicomputadoras. El formato delimitado por un espacio en blanco no es apropiado para una base de datos en la que los campos de carácter puedan incluir espacios, ya que cualquier espacio en blanco dentro del campo sería interpretado como un separador de campo.

Cuando utilice la orden APPEND para introducir un archivo delimitado en una base de datos de dBASE IV, los campos del archivo de texto se transfieren a los campos de la base de datos correspondientes uno a uno; todos los caracteres hasta la primera coma caerán en el primer campo de la base de datos, los caracteres entre la primera y segunda coma caerán dentro del segundo campo, y así sucesivamente. Si existen campos adicionales en el archivo delimitado, el contenido de estos campos se perderá, si existen menos permanecerán vacíos los campos sobrantes de los nuevos registros de la base de datos. Los ítems del archivo delimitado que sean más cortos que la anchura de los campos de dBASE IV correspondientes serán rellenados con espacios en blanco; si los campos son demasiado largos, los datos de carácter se truncan y los campos numéricos son sustituidos por asteriscos para indicar un desbordamiento numérico.

Un archivo de texto de *longitud fija* es un archivo de texto ASCII en el que cada campo ocupa un número fijo de caracteres; cada registro es por tanto de longitud fija. Este formato es también referido en la documentación de dBASE IV como SDF, que es un acrónimo de "System Data Format", un término utilizado solamente por Ashton-Tate para describir este formato. En un archivo de longitud fija (como el archivo .DBF), no hay separadores de campo o delimitadores; cada campo se identifica por su posición inicial en el registro. Si la entrada en un campo es menor que la anchura del campo completo, éste es rellenado con espacios en blanco (los campos de carácter son generalmente justificados a la izquierda dentro de la anchura del campo, mientras los campos numéricos se justifican a la derecha). Como en el formto delimitado, cada registro finaliza con un retorno de carro.

Los archivos de longitud fija pueden ser leídos por procesadores de texto, incluyendo Microsoft Word, y son utilizados por muchos programas de contabilidad y otros gestores de bases de datos. Además, la mayoría de los datos cargados en las bases de datos de los minicomputadores están en un formato de longitud fija (aunque muchos archivos no incluyen el retorno de carro que dBASE IV requiere al final de cada registro). La Figura 26-6 ilustra el archivo SDF producido por una orden COPY similar a la utilizada para ilustrar el formato de archivo delimitado:

```
USE NWCUST ORDER ACCOUNT
COPY TO NWIBM1 TYPE SDF FIELDS Contact, Company, Address1,;
   Address2, City, State, Zip, Lastorder FOR "IBM" $ Equipment
```

Puesto que los campos del archivo SDF están identificados solamente por sus posiciones en el registro, la secuencia y la longitud de los campos comunes al archivo de dBASE IV y al archivo extremo deben ser idénticos cuando copia los datos de dBASE IV en un formato SDF y cuando añade un archivo SDF en una base de datos de dBASE IV. Cuando añade datos desde un archivo SDF, si el registro de longitud fija es mayor que la longitud del registro de la base de datos, los datos de más se pier-

Gina Aronoff		601 First Street	Benicia	CA94510	19881214
Barbara Goddard	Ellis Manufacturing	832 130th N.E.	Bellevue	WA98005	19880216
Lisa Burns	The Image Makers	1900 Powell Street	Emeryville	CA94608	19871112
Carolyn Sumner	J. Thomas Johnson, CPA	50 California Street	San Francisco	CA94111	19881206
Richard Kelly	Kelly and Sons Furniture	14800 Bancroft Avenue	San Leandro	CA94578	19890123
Judy Barnes	Carol Klein, M.D.	1849 S.E. 40th Avenue	Portland	OR97214	19881228
Joan Mills	Lewis and Associates	460 Grand Avenue	Oakland	CA94610	19890120
Kathy McDonald	RapidType Secretarial Svc	2457 Union Street	San Francisco	CA94123	19890310

Figura 26-6.

Un archivo de longitud fija (SDF).

den; si el registro de la base de datos es más largo, los campos de más permanecen en blanco.

Cuando copia datos a cualquier formato de archivo de texto, dBASE IV transfiere las fechas en el *formato YYYYMMDD* (por ejemplo, 19910826 para el 26 de agosto de 1991) y convierte los campos lógicos en una sola letra, T, F, Y y N. Cuando añade datos desde un archivo de texto, dBASE IV asume también estos formatos.

Consideraciones especiales para las hojas electrónicas

De los cuatro formatos propuestos para intercambiar datos con hojas electrónicas, .DIF, SYLK, .WKS y .WK1, solamente se utilizan hoy en día los dos formatos de las hojas electrónicas de Lotus. Aunque puede utilizar el formato SYLK para trasnsferir datos a Excel de Microsoft y Multiplan, Excel también puede leer y escribir hojas de trabajo de Lotus y archivos dBASE III PLUS. Cuando intercambia datos con una hoja electrónica, cada fila de la hoja electrónica corresponde a un registro de la base de datos, y cada columna corresponde a un campo. Si no está familiarizado con el software de las hojas electrónicas, puede imaginar la hoja electrónica como análoga a la pantalla del modo Hojear, con los registros dispuestos horizontalmente y los campos alineados en columnas.

Cuando exporta datos de dBASE IV a un formato de Lotus, dBASE IV crea un nuevo archivo de hoja electrónica Lotus con un registro en cada fila y un campo en cada columna independiente, y escribe el nombre de los campos en la primera fila de la hoja electrónica como encabezamiento de la columna. Puesto que todas las celdas excepto aquéllas que contienen campos numéricos están protegidas, si activa la protección de la hoja electrónica sólo podrá introducir datos en los campos numéricos. Los campos memo no son convertidos, y los campos lógicos se convierten en etiquetas ("Y" de .T. y "N" de .F.). dBASE IV ajusta la anchura de las columnas de la hoja electrónica para que coincida con la anchura de los campos de la base de datos de dBASE IV; en la mayoría de los casos, antes de trabajar con la hoja electrónica deseará ampliar las columnas para añadir espacios entre los ítems de datos adyacentes o estrecharlas para que quepan más datos en la pantalla. La Figura 26-7 ilustra la hoja electrónica de Lotus generada a partir del archivo de Inventario de National Widgets.


```
A1: [W6] 'CATEGORY                                              READY

     A      B               C                       D       E      F       G
 1  CATEGO ARINUDESCRIP                          VENDORCOST    PRICE
 2  RIBBON270-11NEC 7700 Multistrike             NEC       3.20    5.45
 3  RIBBON270-10NEC 7700 Cloth                   NEC       3.20    5.45
 4  RIBBON270-13NEC 3500 Cloth                   NEC       5.50    9.95
 5  RIBBON270-12NEC 3500 Multistrike             NEC       5.50    9.95
 6  RIBBON240-51Epson FX-285, 286                EPSON     9.75   14.95
 7  RIBBON250-30Okidata 80, 81, 82, 83           OKIDAT    1.50    2.95
 8  RIBBON250-31Okidata 84, 94                   OKIDAT    3.00    4.95
 9  RIBBON240-50Epson MX-80, 85, FX-80, 85       EPSON     7.09   12.50
10  RIBBON240-52Epson EX-80, 100, Black          EPSON     3.60    5.95
11  RIBBON240-53Epson EX-80, 100, Color          EPSON     6.50   10.95
12  RIBBON260-20Apple Imagewriter II             APPLE     3.00    5.95
13  FORMS 803-2014-1/2 X 11" Green bar           MOORE    28.50   45.25
14  FORMS 803-2114-1/2 X 11" White               MOORE    24.00   42.50
15  FORMS 803-319-1/2 x 11" White                MOORE    24.50   37.50
16  FORMS 803-309-1/2 x 11" Green bar            MOORE    25.00   37.50
17  FORMS 820-203-1/2 x 1" Labels, 1-up          AVERY    14.50   21.95
18  FORMS 820-243-1/2 x 4" Labels, 4-up          AVERY    39.50   64.95
19  COVER 540-10Dust cover, Apple IIGS           COMCOV   16.50   24.95
20  COVER 540-11Dust cover, Mac Plus             COMCOV   18.50   29.95
10-Jan-89  11:48 AM                                              CAPS
```

Figura 26-7.

Hoja electrónica generada por una orden COPY.

Si desea añadir datos desde un archivo de dBASE IV a una hoja electrónica creada anteriormente, puede copiar los registros en un archivo delimitado y en 1-2-3 utilizando la secuencia de órdenes /File Import Numbers para introducir este archivo en la hoja electrónica, con la esquina superior izquierda del bloque de datos en la posición del puntero o cursor. De forma alternativa, puede exportar datos a una hoja de trabajo independiente e introducir los datos a partir de esta hoja de trabajo en otro con la secuencia de órdenes /File Combine de Lotus.

Cuando añade datos desde una hoja electrónica de Lotus, dBASE IV supone que la base de datos empieza en la esquina superior izquierda de la hoja de trabajo (celda A1). Si utiliza la orden IMPORT, dBASE IV crea una base de datos con un campo para cada columna que contiene datos. Los designadores de las columnas son las letras A, B, C, etc., y se utilizan como nombre de los campos, la anchura de los campos se deducen de la anchura de las columnas de la hoja electrónica, y a todos los campos se les asigna un tipo de datos de carácter. Cuando utiliza APPEND, dBASE IV transfiere los datos de la hoja electrónica a la base de datos activa, una columna por campo. Si la hoja de trabajo tiene más columnas que campos hay en la base de datos, los datos de las columnas de la hoja electrónica adicionales se pierden; si las bases de datos tienen más campos, los campos que no tienen correspondencia en la hoja electrónica permanecerán en blanco.

Observe que también puede convertir los datos de dBASE IV en un formato de hoja electrónica de Lotus utilizando la utilidad Translate de Lotus, que ofrece una importante ventaja: cuando dBASE IV crea una hoja de trabajo, convierte las fechas

en etiquetas (puede que no haya notado esto al principio ya que las etiquetas tienen el mismo formato que las fechas *dd-Mmm-yy* de Lotus estándar), mientras que la utilidad Translate las convierte en un formato de fechas de Lotus ("fechas en series de números"), que la hoja electrónica puede tratar como fechas válidas.

Un programa para copiar datos seleccionados en un formato externo

Si los usuarios de su aplicación de dBASE IV necesitan acceder a informaciones que se encuentran almacenadas en bases de datos creadas con otro software, puede escribir un programa que facilite la transferencia permitiendo que los usuarios elijan la base de datos y el formato del archivo externo, especifiquen los campos que deberían copiarse, elijan la etiqueta de índices que determina la secuencia de registros en el archivo exportado, e introduzcan el criterio de selección ad hoc introduciendo una selección de dBASE IV, como se describió en el Capítulo 19. Cuando escribe programas para una base de datos particular, puede recoger más criterios de selección detallados utilizando un procedimiento tal como CUSTSEL empleado en los programas de impresión de etiquetas descritos en los Capítulos 19 y 20.

Las ventanas que visualizan las listas de selección de los archivos de bases de datos, formatos de archivo externo y campos (DATABASE, FILETYPE y FIELDS, respectivamente) están definidas en un procedimiento llamado MENUDEF. La selección del usuario de cada lista se procesa por un procedimiento que tiene el mismo nombre que la lista. No puede definir listas de selección de etiquetas de índices hasta que el usuario haya seleccionado una base de datos, e incluso después, la lista no puede ser definida explícitamente (el programa debe construir la lista leyendo el nombre de las etiquetas almacenadas en el archivo de producción .MDX).

El programa NWCOPY visualiza la primera lista de selección DATABASE. La selección del usuario se almacena en una variable de memoria llamada MDATABASE, que se inicia en la parte principal del programa a una cadena nula. Si el usuario pulsa ESC para abandonar la lista de selección sin elegir ningún archivo, el programa elimina las ventanas y apariciones súbitas definidas y regresa al programa que llama. Si el usuario selecciona un archivo, el procedimiento DATABASE, que procesa la selección de la lista de selección, almacena el nombre del archivo en MDATABASE, abre el archivo seleccionado, y desactiva la ventana.

Una vez que se abre una base de datos, el programa llama al procedimiento PICKINDEX, que construye la lista de selección de etiquetas de índices según el archivo de producción .MDX. Este procedimiento utiliza un bucle DO WHILE para leer las etiquetas una a una del archivo de producción .MDX, hasta que la función TAG contiene una cadena nula, es decir, hasta que no existen más etiquetas. En cada paso a través del bucle, se define una línea en la ventana que contiene la lista de etiquetas de índice. El número de línea es el valor actual de la variable de memoria MCOUNT, que se incrementa en cada paso a través del bucle. El mensaje es el nombre de la etiqueta actual, deducido por la función TAG, y el mensaje visualiza la expresión clave, que está disponible con la función KEY. Las etiquetas de la lista de selección resultantes se visualizan en el orden en que fueron añadidas al archivo de producción


```
* NWCOPY.PRG
* PROGRAMA PARA COPIAR DATOS SELECCIONADOS EN FORMATOS EXTERNOS
* ULTIMA REVISION 07/18/89 M. LISKIN

CLEAR
DO Screenhead WITH "", "Copia los datos seleccionados en un archivo externo"

* Define las ventanas y las listas de selección
DO Menudef

_wrap = .T.

* Visualiza una lista de selección de las bases de datos
@ 10,10 SAY "Elija la base de datos a copiar"
mdatabase = ""                                  && Nombre de la base de datos
ACTIVATE POPUP database
IF LEN(mdatabase) = 0                           && Si no selecciona una base de
                                                   datos,
   RELEASE POPUP database, filetype, fields
   RETURN                                       &&    sale al menú
ENDIF

* Visualiza una lista de selección de etiquetas de índices
@ 10,10
@ 11,10
@ 10,10 SAY "Seleccione la secuencia de registro"
mindex = ""                                     && Nombre de la etiqueta de índi-
                                                   ces
DO Pickindex                                    && Construye y visualiza la lista
                                                &&     de selección de etiquetas
                                                        de índices

* Visualiza una lista de selección de campos
@ 10,10
@ 10,10 SAY "Seleccione los campos a incluir"
@ 11,10 SAY "(sale inmediatamente al copiar todos los campos)"
_wrap = .T.                                     && Activa la justificación
_rmargin = 45                                   && Margen derecho
mfields = ""                                    && Lista de campo
ACTIVATE POPUP fields

* Si no selecciona ningún campo, convierte la lista para validar la cláusula
  FIELDS
IF LEN(mfields) > 0
   mfields = "FIELDS" + LEFT(mfields, LEN(mfields)-2)
ENDIF
```

Figura 26-8.

Un programa para copiar los datos en un formato de archivo externo.

```
_rmargin = 80
@ 10,0 CLEAR

* Capta el criterio de selección ad hoc
mcondition = SPACE(250)                    && Criterio de selección
@ 10,10 SAY "Introduzca la condición que describe el criterio de selección de
            registro:"
@ 12,10 GET mcondition FUNCTION "S60";
      VALID mcondition = " " .OR. TYPE(mcondition) = "L";
      ERROR "No es una condición de dBASE IV válida"
READ

* Visualiza una lista de selección de tipos de archivos
@ 10,10 CLEAR
@ 10,10 SAY "Seleccione el formato del archivo externo"
STORE "" TO mfiletype, mdescript, mext
ACTIVATE POPUP filetype
IF LEN(mfiletype) = 0
   RELEASE POPUP database, filetype, fields
   RELEASE WINDOW copy
   RETURN
ENDIF

mfilename = SPACE(8)
@ 10,10
@ 10,10 SAY "Introduzca el nombre del archivo externo" GET filename;
      FUNCTION "!" VALID mfilename <> " ";
      ERROR "debe introducir un nombre para el archivo externo"
READ

ACTIVATE WINDOW copy
_lmargin = 2
_rmargin = 58
?  "Base de datos: "
?? mdatabase
?  "Indice: "
?? mindex
?  "Campos: ";
?? SUBSTR(mfields,8) FUNCTION "V56",
?
?  "Selecciones: "
?? TRIM(mcondition)
?
?  "Formato: "
?? mdescrip
?  "Nombre del archivo: "
```

Figura 26-8.

Un programa para copiar los datos en un formato de archivo externo *(continuación)*.


```
?? TRIM(mfilename) + "." + mext
?
mconfirm = .T.
@ 12,10 SAY "¿Está preparado para comenzar la copia? (Y/N)" GET mconfirm;
      PICTURE "Y"
READ
IF mconfirm
   CLEAR
   @ 5,10 SAY "Copiando los datos - Por favor no interrumpa la operación"
   SET FILTER TO &mcondition.
   SET TALK ON
   IF mfiletype = "PFS"
      EXPORT TO (mfilename) TYPE &mfiletype. &mfields.
   ELSE
      COPY (mfilename) TYPE &mfiletype. &mfields.
   ENDIF
   SET TALK OFF
ENDIF

USE
DEACTIVATE WINDOW copy
RELEASE WINDOW copy
RELEASE POPUP database, filetype, fields
RETURN

*****************************************************************************

* Define las ventanas, los menús y las listas de selección
PROCEDURE Menudef

* Define las ventans para visualizar las selecciones y monitorizar la orden
COPY
DEFINE WINDOW copy FROM 6,10 TO 20,70 DOUBLE

* Define la lista de selección de bases de datos
DEFINE POPUP database FROM 5,50 TO 20,65 PROMPT FILES LIKE ".DBF";
   MESSAGE "Destaque el nombre de una base de datos, y pulse <ENTER> para se-
           leccionar"
ON SELECTION POPUP database DO Database

* Define la lista de selección de formatos de archivo
DEFINE POPUP filetype FROM 5,50
DEFINE BAR  1 OF filetype PROMPT " Texto delimitado por comas "
DEFINE BAR  2 OF filetype PROMPT " Texto de longitud fija"
DEFINE BAR  3 OF filetype PROMPT " Lotus 1-2-3"
DEFINE BAR  4 OF filetype PROMPT " Framework II"
```

Figura 26-8.

Un programa para copiar los datos en un formato de archivo externo *(continuación)*.

```
DEFINE BAR  5 OF filetype PROMPT " RapidFile"
DEFINE BAR  6 OF filetype PROMPT " dBASE II"
DEFINE BAR  7 OF filetype PROMPT " dBASE III/III PLUS"
DEFINE BAR  8 OF filetype PROMPT " PFS:FILE"
DEFINE BAR  9 OF filetype PROMPT " VisiCalc (DIF)"
DEFINE BAR 10 OF filetype PROMPT " Multiplan SYLK"
ON SELECTION POPUP filetype DO Filetype

* Define la lista de estructura
DEFINE POPUP fields FROM 5,50 TO 20,65 PROMPT STRUCTURE ;
   MESSAGE "Seleccione los campos pulsando <ENTER>, luego <ESC> para salir"
ON SELECTION POPUP fields DO Fields

RETURN
* Fin del procedimiento Menudef

* Procesa la selección de la base de datos
PROCEDURE Database
mdatabase = PROMPT()
USE (mdatabase)
DEACTIVATE POPUP
RETURN
* Fin del procedimiento Database

* Visualiza la lista de selección de etiquetas de índices
PROCEDURE Pickindex

DEFINE POPUP indextag FROM 5,50;
   MESSAGE "Destaque un índice, y pulse <ENTER> para seleccionarlo"
mcount = 1
DO WHILE LEN(TAG(mcount)) > 0
   DEFINE BAR mcount OF indextag PROMPT TAG(mcount);
      MESSAGE "Indice basado en " + KEY(mcount)
   mcount = mcount + 1
ENDDO
DEFINE BAR mcount OF indextag PROMPT "Sin índice";
   MESSAGE "Copia los registros en orden natural"

ON SELECTION POPUP indextag DO Indextag

ACTIVATE POPUP Indextag
RETURN
* Fin del procedimiento Pickindx

* Procesa la selección de la etiqueta elegida como índice
PROCEDURE Indextag
```

Figura 26-8.

Un programa para copiar los datos en un formato de archivo externo *(continuación)*.


```
mindex = PROMPT()
IF mindex <> "Sin índice"
   SET ORDER TO &mindex.
ENDIF
DEACTIVATE POPUP
RETURN
* Fin del procedimiento Indextag

* Procesa la selección del tipo de archivo
PROCEDURE Filetype

mdescrip = LTRIM(PROMPT())

DO CASE
   CASE BAR() = 1
      mfiletype = "DELIMITED"
      mext = "TXT"
   CASE BAR() = 2
      mfiletype = "SDF"
      mext = "TXT"
   CASE BAR() = 3
      mfiletype = "WKS"
      mext = "WKS"
   CASE BAR() = 4
      mfiletype = "FW2"
      mext = "FW2"
   CASE BAR() = 5
      mfiletype = "RPD"
      mext = "RPD"
   CASE BAR() = 6
      mfiletype = "DB2"
      mext = "DB2"
   CASE BAR() = 7
      mfiletype = "DBMEMO3"
      mext = "DB3"
   CASE BAR() = 8
      mfiletype = "PFS"
   CASE BAR() = 9
      mfiletype = "DIF"
      mext = "DIF"
   CASE BAR() = 10
      mfiletype = "SYLK"
ENDCASE

DEACTIVATE POPUP
RETURN
* Fin del procedimiento Filetype
```

Figura 26-8.

Un programa para copiar los datos en un formato de archivo externo *(continuación)*.

```
* Procesa la selección del campo
PROCEDURE Fields
IF .NOT. PROMPT() $ mfields
   mfields = mfields + PROMPT() + ","
   @ 15,0 SAY ""
   ? "Lista de campos: " + mfields
   ?
ENDIF
RETURN
* Fin del procedimiento Fields

* Fin del programa NWCOPY.PRG
```

Figura 26-8.

Un programa para copiar los datos en un formato de archivo externo *(continuación)*.

.MDX, pero en la mayoría de los casos, el usuario puede localizar las etiquetas de un vistazo.

Cuando finaliza el bucle DO WHILE debido a que no hay más etiquetas y LEN(TAG(mcount)) es cero, el procedimiento define la última línea: "Sin índices", y el mensaje "Copia de los registros en orden natural". Debido a que el programa incrementa MCOUNT dentro del bucle DO WHILE, esta variable almacena el número de línea correcto para la línea "Sin índices" (un número de etiquetas mayor que el último número encontrado en el archivo de producción .MDX), cuando finaliza el bucle. Nótese también que la orden DEFINE POPUP del principio del procedimiento PICKINDEX no especifica el extremo inferior derecho de la lista de selección; dBASE IV determina el tamaño del marco a partir del número y longitud de las líneas.

Aunque la construcción de las listas es bastante complicada, el proceso de la selección del usuario es tan directa como el de cualquier otra lista de selección. El procedimiento INDEXTAG almacena el texto de la línea seleccionada en la variable de memoria MINDEX, y si el usuario no elige "Sin índices" ejecuta una orden SET ORDER para asignar la etiqueta elegida como índice principal. Si la selección de la lista fue "Sin índices", dBASE IV no asigna un índice principal al archivo y la base de datos se procesa en orden natural.

A continuación, el programa visualiza la lista de selección FIELDS, una lista de estructura que incluye todos los campos de la base de datos activa, y permite que el usuario seleccione uno o más campos. Como se dijo en el Capítulo 17, dBASE IV le permite seleccionar solamente un ítem a la vez de la lista de selección. Para evitar esta limitación, el programa NWCOPY.PRG no desactiva la lista de selección inmediatamente después de procesar cada selección; la ventana FIELDS se desactiva solamente cuando el usuario pulsa ESC para indicar que se ha completado la lista de campos y desea salir sin elegir otro campo. El procedimiento FIELDS construye la lista de campos almacenándola en una variable de memoria llamada MFIELDS. Para cada campo seleccionado de la lista de selección, el procedimiento FIELDS comprueba si la selección está presente en MFIELDS, y en caso negativo, añade el campo a la lista seguido por una coma y un espacio (el espacio opcional se añade principalmente para mejorar la apariencia de la lista de campos visualizada en la pantalla).

También es una buena idea visualizar la lista de campos actual después de cada selección para recordar al usuario los campos que se han seleccionado ya. Para hacer esto sin escribir encima de la lista de selección visualizada en la pantalla, el programa establece el margen derecho en 45 y _WRAP en .T. antes de visualizar la lista de estructura. El procedimiento FIELDS utiliza una orden @... SAY para posicionar el cursor en la fila 15 de la pantalla y una orden ? para visualizar la lista de campos en la mitad izquierda de la pantalla.

Cuando se desactiva la ventana FIELDS (debido a que el usuario ha pulsado ESC en lugar de elegir otro campo), la parte principal del programa comprueba la longitud de MFIELDS para determinar si el usuario ha elegido al menos un campo; en caso contrario, la variable mantendrá su valor nulo inicial. Si MFIELDS contiene una lista de nombres de campo, el programa construye una cláusula FIELDS sintácticamente correcta y almacena esta cláusula en MFIELDS. La cláusula FIELDS consta de la palabra clave "FIELDS" seguida por un espacio y el resultado de extraer los dos últimos caracteres (la coma final, y el espacio añadido por el procedimiento FIELDS) de la lista creada por el procedimiento FIELDS.

Los restantes campos son fáciles. El programa inicializa una variable de memoria de 250 caracteres de longitud para almacenar el criterio de selección ad hoc y permitir que el usuario introduzca cualquier expresión lógica legítima en esta variable. La cláusula VALID de la orden @... GET que capta MCONDITION utiliza la función TYPE para verificar que la entrada del usuario es una expresión lógica válida. Además, el programa acepta la condición si el usuario deja la entrada vacía para evitar este paso (si MCONDITION está vacía, la orden SET FILTER ejecutada antes de empezar la operación de copia no tendrá efecto).

A continuación, el programa activa la ventana FILETYPE para visualizar la lista de formatos de archivos externos. El procedimiento FILETYPE, que procesa la selección de la lista, utiliza una estructura DO CASE que almacena la palabra clave que debe añadirse a la orden COPY para crear un archivo del tipo designado en la variable de memoria MFILETYPE. La descripción del formato externo, el texto del mensaje, se mantiene en la variable de memoria MDESCRIP de modo que el programa pueda visualizar más tarde una advertencia de que el formato ya ha sido seleccionado, y la extensión por omisión que dBASE IV asignará a este archivo se almacena en MEXT para visualizar el nombre completo del archivo que será creado. Finalente, el programa pide al usuario el nombre del archivo exportado (la extensión la proporciona automáticamente dBASE IV basándose en MFILETYPE).

Antes de empezar a copiar los datos, el programa activa la ventana COPY definida anteriormente por el procedimiento MENUDEF, recapitula las selecciones del usuario visualizando las variables de memoria creadas para almacenar las selecciones de la lista, y pide confirmación para proceder. En la Figura 26-9 se muestra una pantalla típica. Si el usuario responde "Y" a la pregunta "¿Está preparado para empezar la copia?", el programa ejecuta la orden SET FILTER que implementa el criterio de selección y copia los datos utilizando una orden COPY para los tipos de archivos externos o una orden EXPORT si el tipo de archivo es PFS (el único formato que no

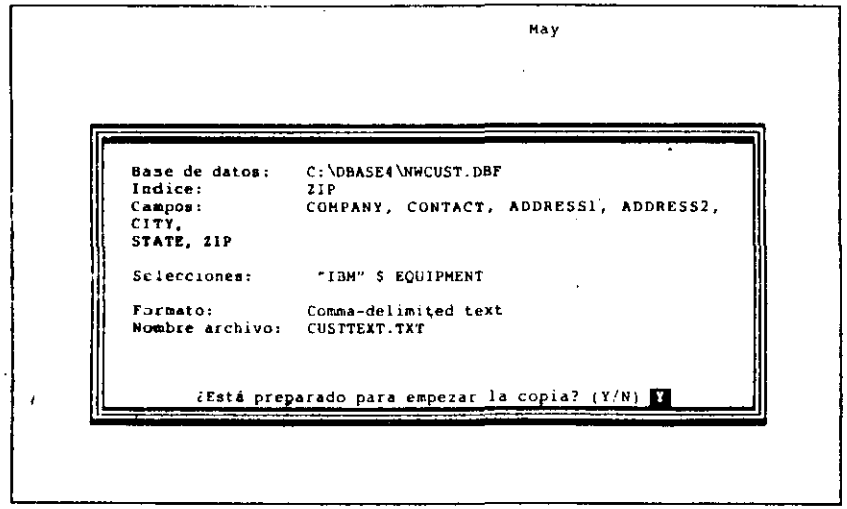

Figura 26-9.

Uso del programa de copia de archivos para crear un archivo delimitado.

está soportado por COPY). El nombre del archivo, tipo de archivo, y lista de campos se expresan extendiendo las variables de memoria correspondientes con el símbolo de sustitución de macro. Si la base de datos seleccionada es extensa, la operación de copia puede llevar algún tiempo y el programa pone el parámetro TALK en ON para permitir que el usuario monitorice el progreso del programa.

Compensación de las diferencias en la estructura del archivo

El programa descrito en la sección anterior supone que la relación existente entre los campos de dBASE IV y los items de datos de un archivo externo es univoca y que los tipos de datos y formatos de visualización son compatibles. Si no es este el caso, debe ajustar los datos de dBASE IV para acomodarlos a los requerimientos del formato externo (a menos que los usuarios estén experimentados en el tratamiento de datos con programas externos y estén dispuestos a hacer las conversiones necesarias sobre los datos exportados). Podría disponer de campos extensos, campos acortados, campos combinados, campos divididos, o convertir un campo a un nuevo tipo de datos. Por ejemplo, puede que necesite exportar los campos nombre y dirección del archivo Clientes a un archivo delimitado en el que la ciudad, estado y código postal se almacenan como un único item, y el área de código y número de teléfono se combinan en la forma 415/848-0121.

Una utilidad de copia de archivos de uso general tal como NWCOPY.PRG no

trata tales transformaciones de datos en detalle, y en la mayoría de los casos debe escribir un programa a medida para llevar a cabo la conversión. Si el formato del archivo externo requerido es uno de los formatos estándar soportados por dBASE IV, la forma más fácil de transferir los datos es crear un archivo intermedio que coincida con la disposición de campos del archivo externo, convertir los datos de dBASE IV a este formato, y luego utilizar una orden COPY o EXPORT normal para exportar los datos. La Figura 26-10 lista la estructura de una base de datos temporal llamada NWCUST2.DBF, que se puede utilizar para convertir el archivo Clientes al formato descrito en el ejemplo anterior. Para realizar la conversión puede utilizar el siguiente programa, que examina el archivo Clientes secuencialmente y añade a la base de datos NWCUST2.DBF un registro que coincide con cada uno de los registros de clientes:

```
USE NWcust ORDER Account ALIAS Customer
USE NWcust IN 2 ALIAS Transfer

SCAN
   SELECT Transfer
   APPEND BLANK
   REPLACE Contact WITH Customer->Contact, Company WITH Customer->Company,;
      Address1 WITH Customer->Address1, Address2 WITH Customer->Address2,;
      City WITH TRIM(Customer->City) + "," + Customer->State + " " +;
      Customer->Zip, Telephone WITH Customer->Areacode + "/" +;
      Customer->Telephone
   SELECT Customer
ENDSCAN

CLOSE DATA
```

Después puede abrir NWCUST2.DBF y utilizar una orden EXPORT o COPY para crear el archivo externo. Si necesita representar los datos desde más de un archivo,

```
Estructura de la base de datos: C:\DBASE4\NWCUST2.DBF
Número de registros:     0
Fecha de última actualización   :  08/09/89
Campo    Nombre     Tipo       Ancho  Dec    Indi.
    1    COMPANY    Carácter      25           N
    2    ADDRESS1   Carácter      25           N
    3    ADDRESS2   Carácter      25           N
    4    CITY       Carácter      35           N
    5    TELEPHONE  Carácter      12           N
    6    CONTACT    Carácter      25           N
** Total **                      148
```

Figura 26-10.

Archivo intermedio utilizado para exportar los datos del nombre y la dirección del cliente.

```
SCAN
   ?? Contact
   ?? Company
   ?? Address1
   ?? Address2
   ?? LEFT(TRIM(City) + "," + State + " " + Zip + SPACE(35), 35)
   ?? Areacode + "/" + Telephone
ENDSCAN

CLOSE ALTERNATE
USE
```

En un archivo de longitud fija, un campo dado debe ser de la misma longitud en cada registro. Para combinar la ciudad, estado, y código postal en un formato estándar sin producir registros de longitud variable, este programa añade 35 espacios a la combinación ciudad/provincia/ZIP y luego utiliza la función LEFT para extraer los 35 primeros caracteres de la cadena resultante. Si la estructura del registro está dictada inequívocamente por los requerimientos del programa que usará los datos exportados, puede que necesite ajustar la longitud de los campos añadiendo espacios a algunos campos y utilizando la función LEFT para visualizar solamente una parte de los otros. Nótese también que para producir la salida no formateada requerida, debe establecer _WRAP en .F. para evitar que dBASE IV añada retornos de carro para justificar la salida y acomodarla a los márgenes actuales.

PROGRAMAS DE VALIDACION DE DATOS SECUENCIALES

Los programas presentados hasta ahora han validado datos que han sido introducidos utilizando cláusulas VALID, funciones definidas por el usuario, y consultas en otras bases de datos. A veces, puede que desee captar datos sin validarlos y realizar pruebas de validación en un proceso secuencial más tarde. El mejor momento para realizar pruebas de validación depende en parte de las consecuencias de introducir datos erróneos en el archivo y en su comprensión de los parámetros que utilizarán los programas. La validación secuencial es a veces apropiada en situaciones en las que se introducen gran cantidad de datos o a gran velocidad, es decir, entradas iniciales de cientos o miles de registros en un sistema de base de datos nuevo, y es esencial para comprobar los datos procedentes de una minicomputadora o datos convertidos a partir de un archivo creado por otro programa. Mientras que los datos convertidos generalmente requieren una corrección cuidadosa para captar los errores, un programa de validación secuencial puede ser una eficiente protección en primera instancia para asegurar que la mayoría de las entradas sean razonables.

Las pruebas que necesita realizar en una base de datos dada son a menudo altamente personalizadas a los requerimientos de una organización particular. En el sistema National Widgets, por ejemplo, pueden definirse las siguientes reglas para los datos introducidos en el archivo Clientes:

- El campo ACCOUNT debe ser único y no debe estar vacío.
- Los campos COMPANY y CONTACT no deben estar vacíos (debe introducirse el nombre de compañía o el nombre del contacto para identificar al cliente).
- La primera línea de dirección no debe estar vacía (ADDRESS1).
- El campo CITY no debe estar en blanco.
- El campo STATE debe contener una abreviatura postal del estado válida.
- El campo ZIP debe contener un código numérico válido de 5 ó 10 dígitos.
- El campo AREACODE debe estar en blanco o debe contener un código de área legítimo; no obstante, no puede estar en blanco si se ha cumplimentado el número de teléfono.
- La primera fecha de pedidos no puede ser posterior a la fecha actual.
- La fecha del último pedido no puede ser anterior a la primera o posterior a la fecha actual.

Estas reglas son típicas de las pruebas de validación aplicadas generalmente al nombre y la dirección de las bases de datos. Ya sabe cómo llevar a cabo la mayoría de estas pruebas, ya que en los archivos de formato y en los programas de entrada de datos descritos en los capítulos anteriores se han incorporado similares o idénticas validaciones. Cuando diseñe un sistema, se dará cuenta que la definición clara de las reglas de validación traña normalmente más dificultad que la propia implementación de las pruebas, en parte debido a que los usuarios puede que tengan dificultades en precisar los términos, y es posible que la comprensión de las reglas requiera algún conocimiento acerca de la organización para la que está escribiendo los programas.

La estrategia básica para cualquier programa secuencial de validación es examinar la base de datos, aplicar las condiciones a cada uno de los registros, y anotar o imprimir los fallos de cualquiera de ellas. Debido a que esta estrategia puede llevar algún tiempo para procesar un archivo extenso, generalmente es mejor imprimir una lista de los registros que no cumplen las condiciones en lugar de detenerse cuando se encuentra el problema y pedir que el operador lo corrija. La impresión del listado de errores le permite ejecutar el programa ininterrumpidamente, y da a los usuarios tiempo para localizar la información errónea o incorrecta antes de volver a actualizar el programa para determinar el problema.

Si un registro pasa todas las pruebas de validaión, no debería imprimirse, y si falla en uno o más criterios, solamente deberían aparecer en el listado de errores los campos no válidos. Para hacer que el informe sea lo más informativo posible, el programa imprimirá el contenido de los campos incorrectos y quizá una breve descripción del error en los datos. Por ejemplo, si el campo STATE estaba en blanco, podría imprimir "Estado en blanco" o "Estado incorrecto". Para una abreviatura del estado no válida, el programa podría imprimir el campo STATE erróneo, seguido por una explicación tal como "Esta abreviatura del estado no es válida".

La Figura 26-11 lista un programa secuencial de validación para el archivo Clientes de National Widgets, NWCVALID.PRG. El programa abre el archivo de Clien-

```
* MWCVALID.PRG
* PROGRAMA DE VALIDACION SECUENCIAL PARA EL ARCHIVO DE CLIENTES
* ULTIMA REVISION: 08/04/89 M. LISKIN

USE NWcust ORDER Account ALIAS Customer
USE NWcodes IN 2 ORDER Codetype ALIAS Codes

DO Scrnhead WITH "", "Valida el archivo Clientes"

* Asegura que la impresora está en línea
DO WHILE .NOT. PRINTSTATUS()
   DO Message1 WITH "Por favor conecte la impresora y asegúrese de que está en
                    línea" +;
      "y preparada para imprimir el listado de los errores"
ENDDO

* Establece la impresión del informe
SET PRINT ON
SET CONSOLE OFF
_wrap = .T.
ON PAGE AT LINE 60 DO Pagebrk
DO Pagehead                          && Cabecera de página en la primera página

DO Monitor WITH "Valida los registros de clientes"

maccount = Account                   && Código de cuenta del registro
                                     &&    anterior

SCAN

   @ 4,28 SAY RECNO()

   MPRINTED = .F.                    && El registro no se ha impreso todavía

   IF Account = " "
      DO Errormsg WITH "El código de cuenta está en blanco"
   ELSE
      IF Account = maccount
         DO Errormsg WITH "El código de cuenta está duplicado"
      ENDIF
   ENDIF

   IF Company = " " .AND. Contact = " "
      DO Errormsg WITH "El nombre de la compañía y el nombre del contacto es-
                       tán en blanco"
   ENDIF
```

Figura 26-11.

Programa de validación secuencial para el archivo Clientes.


```
   IF Address1 = " "
      DO Errormsg WITH "La primera dirección está en blanco"
   ENDIF

   IF City = " "
      DO Errormsg WITH "La ciudad está en blanco"
   ENDIF

   IF .NOT. SEEK("S"+State, "Codes")
      DO Errormsg WITH State + " no es una abreviatura de estado válida"
   ENDIF

   IF .NOT. Zipok(Zip, "")
      DO Errormsg WITH Zip + " no es un código postal válido"
   ENDIF

   IF Areacode = " "
      IF Telephone <> " "
         DO Errormsg WITH;
            "El código del área es incorrecto, aunque está cumplimentado el
             número de teléfono"
      ENDIF
   ELSE
      IF .NOT. SEEK("A"+Areacode, "Codes")
         DO Errormsg WITH Areacode + " no es un código de área válido"
      ENDIF
   ENDIF

   IF Firstorder > DATE()
      DO Errormsg WITH "La fecha del primer pedido es posterior a la fecha de
                       hoy"
   ENDIF

   IF Lastorder > DATE()
      DO Errormsg WITH "La fecha del último pedido es posterior a la fecha de
                       hoy"
   ENDIF

   IF Lastorder < Firstorder
      DO Errormsg WITH "La fecha del último pedido es anterior a la fecha del
                       primer pedido"
   ENDIF

   maccount = Account

ENDSCAN
```

Figura 26-11.

Programa de validación secuencial para el archivo Clientes *(continuación)*.

```
EJECT PAGE
SET PRINT OFF .
ON PAGE
DEACTIVATE WINDOW monitor
CLOSE DATABASES
RETURN

*************************************************************************

* Manipulador de división de página
PROCEDURE Pagebrk
EJECT PAGE
IF .NOT. EOF()
   DO Pagehead
ENDIF
RETURN
* Fin del procedimiento Pagebrk

* Imprime la cabecera de página
PROCEDURE Pagehead
?
?
DO Center2 WITH 80, "BU", "Registros de clientes con fechas no válidas"
?
?
? "Registro   Cantidad   Problema"
? REPLICATE("-",80)
?
RETRURN
* Fin del procedimiento Pagehead

* Imprime un mensaje de error

PROCEDURE Errormsg
PARAMETER mmessage
IF .NOT. mprinted
   ?  RECNO() PICTURE "999999"
   ?? Account AT 10
   mprinted = .T.
ENDIF
?? mmessage AT 25
?
RETURN
* Fin del procedimiento Errormsg

* Fin del programa NWCVALID.PRG
```

Figura 26-11.

Programa de validación secuencial para el archivo Clientes *(continuación)*.

tes y el archivo de Códigos (que se requiere para validar los campos STATE y AREACODE). Aunque la lectura del archivo de Clientes en orden natural daría por resultado un proceso rápido, el programa asigna la etiqueta ACCOUNT como índice principal por dos razones: en primer lugar, la condición para los códigos de cuenta duplicados dependen de la lectura de la base de datos en orden indexado de modo que los registros con valores idénticos en el campo ACCOUNT están agrupados secuencialmente. Además, la impresión de una lista de errores en orden alfabético es más conveniente para las personas que tienen problemas en localizar los errores y corregirlos.

El programa de validación utiliza una llamada explícita a la función PRINTSTATUS para verificar si la impresora está conectada y preparada para recibir los datos, en lugar de confiar en el programa de captación de errores descrito en el Capítulo 25 para detectar y responder al mensaje de error "La impresora no está preparada" ("Printer not ready"). Esta advertencia es especialmente importante ya que, a diferencia de otras opciones de menú del sistema National Widgets que imprimen informes, puede que no esté inicializada y el programa de validación generará cualquier salida impresa y puede que el mensaje de error general visualizado por NWERROR.PRG le parezca irrelevante o no apropiado.

Antes del bucle SCAN que procesa el archivo de Clientes, el programa de validación llama al procedimiento MONITOR para establecer el estado de visualización y permitir que los usuarios sigan la pista del progreso (la orden @... SAY que visualiza el número de registro actual en la posición apropiada en la ventana MONITOR se localiza dentro del bucle). Para preparar la impresión del informe, el programa establece _WRAP en .T. para formatear los mensajes de error correctamente en las salidas impresas, establece una rutina ON PAGE que produce una división de página en la línea 60, y llama al procedimiento PAGEHEAD para producir un encabezamiento de página en el listado de errores (en este caso una sola línea de descripción del informe y un conjunto de cabeceras de columna).

Luego el programa comprueba los códigos de cuenta duplicados comparando el campo ACCOUNT de cada registro del cliente con el valor del campo del registro anterior, que se almacena en la variable de memoria MACCOUNT. Esta variable se inicializa antes de la sentencia SCAN con el valor del campo ACCOUNT del primer registro procesado. En la parte inferior del bucle SCAN, MACCOUNT toma el valor del campo ACCOUNT del registro actual para preparar la prueba del siguiente registro. De un par de registros con el mismo código de cuenta, solamente se imprimirá el segundo en la lista de error, pero cuando los usuarios regresan al programa de consulta y actualización para corregir el problema, la búsqueda de clave de índice basada en el campo ACCOUNT también recuperará el primero. En la mayoría de los casos, será necesario examinar o imprimir ambos registros a la vez para decidir si representan entradas duplicadas del mismo cliente (en cuyo caso debería eliminarse uno) o diferentes clientes a los cuales se les ha asignado un mismo número de cuenta (en cuyo caso, debería modificarse uno de ellos).

Dentro del bucle SCAN, el programa lleva a cabo una serie de tests sobre cada registro e imprime un mensaje de error para cada prueba de validación en la que el registro falla. Para identificar el registro, éste imprime el número de registro y el código de cuenta pero esta información solamente debería aparecer una vez, sin tener

en cuenta el número del mensaje de error asociado con el registro. El programa utiliza una variable de memoria llamada MPRINTED para seguir la lista de si ya se ha impreso el registro. Esta variable se inicializa con el vlaor .F. al principio del bucle SCAN y se vuelve a establecer en .T. después de imprimir el primer mensaje de error.

Para evitar la redundancia de código, la impresión real se lleva a cabo por el procedimiento ERRORMSG. La primera vez que se llama a este procedimiento para un registro de clientes particular, MPRINTED es .F. y el procedimiento imprime el número de registro y el campo ACCOUNT y vuelve a establecer MPRINTED en .T. La siguiente vez que el procedimiento es llamado para el mismo registro de clientes, MPRINTED es .T., la condición presente en la sentencia IF es .F., y el procedimiento sólo imprime el mensaje de error. Después de imprimir cada mensaje de error, el procedimiento ejecuta una orden ? para avanzar el papel a la siguiente línea. Si el procedimiento es llamado de nuevo para el mismo cliente, esta orden ? garantiza que el siguiente mensaje de error aparezca en una nueva línea; si se llama para un nuevo cliente, la orden ? que imprime el número de registro genera otro retorno de carro, y el programa salta una línea entre registros.

ESCRITURA DE UTILIDADES BATCH PARA ACTUALIZAR BASES DE DATOS

La mayoría de las utilidades de mantenimiento de bases de datos descritas hasta ahora en este capítulo funcionan en dBASE IV sin tener en cuenta su contenido; de hecho, el objetivo de diseño principal ha sido hacer que estas utilidades sean aplicaciones tan independientes como sea posible. La mayoría de las aplicaciones de dBASE IV requieren programas de utilidad que manipulen los *datos* almacenados en los archivos de forma específica y lleven a cabo procesos secuenciales que afecten a muchos o a todos los registros del archivo. Por ejemplo, puede que necesite poner a cero varios campos de totales al final del mes o del año, archivar y eliminar datos obsoletos, reconstruir totales en un archivo (tal como el archivo Clientes) en base al contenido de otros archivos almacenados (tal como el archivo Histórico de Transacciones), o cambiar el valor de un campo clave de índice, tal como el código de cuenta del cliente, en todos los registros almacenados en el sistema. La versión de la aplicación de National Widgets diseñada en el Capítulo 11 con el generador de aplicaciones incluyó algunas de las utilidades de mantenimiento de bases de datos localizadas en su menú **Mantenimiento**.

Algunos de estos procedimientos, tal como el **Proceso fin de año**, podrían incorporarse en la versión final de una aplicación con pequeños cambios que mejoren la estética de los mensajes de estado visualizados por el mismo. Otras opciones, tales como **Archivar clientes inactivos**, podrían hacerse más flexibles utilizando las técnicas descritas en el Capítulo 19 para permitir que los usuarios especifiquen los criterios de selección, en este caso, la fecha límite que determina los clientes que se deben archivar o eliminar. No obstante, ésta y otras opciones serían más beneficiosas que una reposición general más sustancial. Recuerde lo difícil que fue depurar el archivo de Transacciones utilizando solamente las órdenes incorporadas en dBASE IV, que no

ofrecen forma de comparar las transacciones relacionadas en la misma base de datos. Con un programa a medida, esta operación es más directa.

La Figura 26-12 lista NWPURGE.PRG, un programa de depuración y almacenamiento de archivos integrado para el sistema National Widgets. Este programa opera sobre el archivo de Clientes y los dos archivos Históricos, pero puede ampliarse fácilmente para incluir un archivo de Transacciones y el archivo de Pedidos actual. Para ilustrar la técnica que identifica las transacciones de factura cerradas, este programa supone que *todas* las facturas y transacciones de pago, las actuales y las cerradas, están presentes en el archivo Histórico de Transacciones y depura solamente las facturas que han sido pagadas completamente, sin tener en cuenta la fecha de cese.

```
* NWPURGE.PRG
* PROGRAMA PARA DEPURAR LOS ARCHIVOS DE CLIENTES E HISTORICOS DE
*   PEDIDOS Y TRANSACCIONES POR FECHA
* ULTIMA REVISION: 08/06/89 M. LISKIN

SET DELETED OFF

CLEAR
DO Scrnhead WITH "", "Depura los archivos por fecha"

DEFINE WINDOW utility FROM 5,10 TO 20,70
ACTIVATE WINDOW utility

mcustdate = DATE() - 730        && Fecha de cese del cliente
mtrandate = DATE() - 365        && Fecha límite de la transacción
mordpurge = .T.                 && ¿Depura los pedidos?
mcustarch = "NWCARCH"           && Archivo de almacenamiento de clientes
mtranarch = "NWTARCH"           && Archivo de almacenamiento de transacciones
mordarch  = "NWDARCH"           && Archivo de almacenamiento de pedidos
mconfirm  = .F.

* Capta las fechas límites
@ 2, 5 SAY "Fecha límite para el último pedido del cliente" GET mcustdate
@ 3, 5 SAY "Fecha límite para el pago completo de la factura" GET mtrandate
@ 4, 5 SAY "¿Depurar el archivo histórico de pedidos? (Y/N)" GET mordpurge;
       PICTURE "Y"
READ

* Copia el nombre de los archivos
@ 7, 5 SAY "Nombre para el archivo que guardará los clientes" GET mcustarch;
       FUNCTION "!"
@ 8, 5 SAY "Nombre para el archivo que guardará las transacciones" GET
           mtranarch;
       FUNCTION "!"
```

Figura 26-12.

Programa de depuración y almacenamiento de archivos.

```
@ 9, 5 SAY "Nombre para el archivo que guardará los pedidos" GET mordarch;
       FUNCTION "!"
ENDIF
READ

* Pide confirmación antes de proceder
@ 12, 5 SAY "¿Está seguro que desea depurar los archivos? (Y/N)";
       GET mconfirm PICTURE "Y"
READ

IF .NOT. mconfirm
   DEACTIVATE WINDOW utility
   RELEASE WINDOW utility
   RETURN
ENDIF

* Abre las bases de datos
USE NWcust ORDER Account ALIAS Customer
USE NWthist IN 2 ORDER Account ALIAS Transact
IF mordpurge
   USE NWohist IN 3 ORDER Account ALIAS Order
   SELECT Customer
   SET RELATION TO Account INTO Transact, Account INTO Order
ELSE
   SET RELATION TO Account INTO Transact
ENDIF

CLEAR
@ 5, 5 SAY "Trabajando con"

SCAN
   @  5,16 SAY Account + "-" + LEFT(Company,15)

   * Borra el cliente si la fecha del último pedido es anterior a la fecha lí-
     mite
   IF Lastorder < mcustdate
      DELETE                              && Borra el cliente
      SELECT Transact                     && Borra las transacciones
      DELETE WHILE Account = Customer->Account
      IF mordpurge
         SELECT Order                     && Borra los pedidos
         DELETE WHILE Account = Customer->Account
      ENDIF
      SELECT Customer
      LOOP
   ENDIF
```

Figura 26-12.

Programa de depuración y almacenamiento de archivos *(continuación)*.


```
   * Procesa las transacciones
   @ 10,15 SAY "Factura"
   SELECT Transact
   SCAN WHILE Account = Customer->Account
      @ 10,25 SAY Invoice
      minvoince = Invoice                 && Registro de la factura actual
      minvbal = 0.00                      && Balance neto de la factura
      mrecord = RECNO()                   && Primer registro de la factura
      mkeep = .F.                         && ¿Mantiene la transacción?
      SCAN WHILE Invoice = minvoice
         DO CASE
            CASE Txntype = "I"            && Transacción de la factura
               IF Date > mtrandate        && Si la factura es posterior a
                                             la fecha limite
                  mkeep = .T.             &&    mantiene las transacciones
                  EXIT
               ELSE
                  minvbal = minvbal + Ammount
               ENDIF
            CASE Txntype = "P"            && Transacción de pago
               minvbal = minvbal - Ammount
         ENDCASE
      ENDSCAN
      IF minvbal = 0.00 .AND. .not. mkeep && Si el balance de la factura es 0
         GOTO mrecord                     &&    y no está marcado,
         DELETE WHILE Invoice = minvoice  && Marca las transacciones
         IF mordpurge                     && Marca los pedidos
            SELECT Order
            SEEK Transact->Account + minvoice
            DELETE WHILE Invoice = minvoice
            SELECT Transact
         ENDIF
      ENDIF
   ENDSCAN
   SELECT Customer
ENDSCAN

CLOSE DATABASES
SET TALK ON

* Almacena los registros marcados
CLEAR
USE NWcust
@ 5,10 SAY "Almacenando los clientes"
COPY TO (mcustarch) FOR DELETED()
@ 5,10 SAY "Empaquetando y reindexando el archivo Clientes"
PACK
```

Figura 26-12.

Programa de depuración y almacenamiento de archivos *(continuación)*.

```
CLEAR
USE NWthist
@ 5,10 SAY "Almacenando las transacciones"
COPY TO (mtustarch) FOR DELETED()
@ 5,10 SAY "Empaquetando y reindexando el archivo de Transacciones"
PACK

IF mordpurge
   CLEAR
   USE NWohist
   @  5,10 SAY "Almacenando los pedidos"
   COPY TO (mordarch) FOR DELETED()
   @  5,10 SAY "Empaquetando y reindexando el archivo Pedidos"
   PACK
ENDIF

USE
DEACTIVATE WINDOW utility
RELEASE WINDOW utility
RETURN

* Fin del programa NWPURGE.PRG
```

Figura 26-12.

Programa de depuración y almacenamiento de archivos *(continuación)*.

El programa NWPURGE.PRG elimina los clientes cuyos últimos pedidos fueron anteriores a la fecha de cese especificada por el usuario junto con todas las transacciones relacionadas y los pedidos opcionales (National Widgets puede mantener los pedidos para un análisis de la evolución de las ventas del producto incluso si un cliente ha sido eliminado). Además elimina los pagos totales de las transacciones de factura antes de una fecha límite especificada por el usuario que coincide con las transacciones de pago, y a discreción del usuario, los artículos de la línea de pedidos.

En primer lugar, el programa NWPURGE.PRG pide al usuario que introduzca las fechas de cese o límites y el nombre de los archivos que almcenarán los datos obsoletos y que confirme si desea o no depurar el archivo Históricos de Pedidos. Después de responder afirmativamente, el programa abre las tres bases de datos y enlaza los archivos Históricos relevantes con el archivo de Clientes con la orden SET RELATION. Si el usuario decide depurar el archivo Histórico de Pedidos, el programa utiliza una orden SEEK explícita para encontar los registros de pedidos que coinciden con cada transacción de factura.

El bucle SCAN más externo del programa examina el archivo Clientes por el código de cuenta. Si la fecha del último pedido es anterior a la fecha límite especificada, el programa marca el registro del cliente y todos los registros de transacciones correspondientes y regresa al principio del bucle SCAN. En cualquier otro caso, selecciona el área de trabajo de la transacción y establece otro bucle SCAN para examinar las transacciones del cliente actual. Puesto que la etiqueta de índices ACCOUNT está basada en los acompos ACCOUNT e INVOICE, el programa de depuración examina las transacciones de clientes agrupadas por facturas.

Para cada grupo, el programa inicializa las variables de memoria en las que se almacenan el número de la factura, el balance neto de la factura, y el número de registro del primer registro del grupo. La variable lógica MKEEP, inicializada a .F., almacena el estado general del grupo de transacciones; esta variable se establece en .T. en el bucle SCAN siguiente si el programa determina que debería mantenerse el grupo. El siguiente bucle SCAN anidado examina el grupo de transacciones y comprueba el campo TXTYPE en una estructura DO CASE. Si el registro representa una factura, el programa compara la fecha de la factura con la fecha límite y sale del bucle si la fecha de la factura es posterior. Estableciendo la variable MKEEP en .T., el programa asegura que la transacción de grupo no será marcada para su eliminación. Si la fecha de la factura es anterior a la fecha límite, el importe de la factura se añade en el total almacenado en la variable de memoria MINVBAL. Para las transacciones de pago, el programa *sustrae* el campo AMOUNT de MINVBAL.

Cuando finaliza el bucle SCAN, el programa comprueba MINVBAL y MKEEP. Si el balance de la factura almacenado en MINVBAL es cero y el programa no activa la señal MKEEP (la factura es anterior a la fecha límite), el programa regresa al primer registro del grupo y elimina todo el grupo (y opcionalmente, los registros de pedidos correspondientes). Esta secuencia de órdenes, aunque algo compleja, es más eficiente en la forma de procesar las bases de datos que las órdenes secuenciales utilizadas en el procedimiento de depuración descrito en el Capítulo 11. Después de marcar todos los registros que deberían eliminarse de las tres bases de datos, el programa las cierra y, en consecuencia, deshace los enlaces, las abre de una en una, copia los registros marcados en el archivo de almacenamiento correspondiente, y luego empaqueta el archivo.

El último ejemplo de este capítulo ilustra una técnica para permitir que el usuario cambie el contenido del campo clave de índice (el código de cuenta del cliente en todas las bases de datos relacionadas del sistema). Las versiones finales del programa de consulta y actualización presentado en el Capítulo 24 no permitían los enlaces existentes entre los registros del cliente y cualquier registro relacionado de los archivos de Transacciones, Pedidos, y el correspondiente archivo Histórico. Para cambiar un código de cuenta de forma segura, debe escribir un programa que modifique el contenido del campo ACCOUNT en todos los registros relacionados de estos archivos. La Figura 26-13 lista un programa denominado NWCHGACT.PRG que abre todas las bases de datos relevantes y lleva a cabo esta operación. Si lo prefiere, podría añadir las órdenes requeridas al programa de consulta y actualización de clientes.

El programa NWCHGACT.PRG abre todos los archivos que incluyen un campo ACCOUNT y establece un bucle "indefinido" con DO WHILE .T. para permitir que el usuario introduzca los códigos de cuenta del cliente de forma repetida. Los nuevos códigos de cuenta y los originales son captados en las variables de memoria MOLDACCT y MNEWACCT. El programa valida ambos códigos de cuenta buscando una coincidencia en el archivo de Clientes; el código anterior es considerado válido si la función SEEK recupera un registro coincidente, y el nuevo código es válido sólo si la función SEEK falla, indicando que el nuevo código de cuenta propuesto no

```
* NWCHGACT.PRG
* PROGRAMA PARA CAMBIAR EL CODIGO DE CUENTA DEL CLIENTE EN TODOS LOS ARCHIVOS
  RELACIONADOS
* ULTIMA REVISION: 08/26/89   M.LISKIN

SET DELETED OFF

CLEAR
DO Scrnhead WITH "", "Cambia los códigos de cuenta"

DEFINE WINDOW utility FROM 5,10 TO 20,70
ACTIVATE WINDOW utility

* Abre las bases de datos
USE NWcust ORDER Account ALIAS Customer
USE NWtxn IN 2 ORDER Account ALIAS Transact
USE NWorder IN 3 ORDER Account ALIAS Order
USE NWthis IN 4 ORDER Account ALIAS Tranhist
USE NWohist IN 5 ORDER Account ALIAS Ordhist

DO WHILE .T.

   moldacct = SPACE(10)                    && Código de cuenta original
   mnewacct = SPACE(10)                    && Nuevo código de cuenta
   mconfirm = .F.                          && Confirmación para proceder

   @ 2,10 SAY "Código de cuenta original" GET moldacct FUNCION "!";
          MESSAGE "Deje el código de cuenta en blanco para salir";
          VALID moldacct = " " .OR. SEEK(moldacct, "Customer");
          ERROR moldacct + "no se encuentra en el archivo Clientes"
   @ 4,10 SAY    Nuevo código de cuenta" GET mnewacct FUNCTION "!";
          MESSAGE "Deje el código de cuenta en blanco para salir";
          VALID mnewacct = " " .OR. .NOT. SEEK(mnewacct, "Customer");
          ERROR mnewacct + "ya está asignado a otro cliente"
   READ

   IF moldacct = " " .OR. mnewacct = " "   && Si cualquier código de cuenta
                                           &&    está en blanco,
                                           &&    sale del programa
   ENDIF

   @ 10,5 SAY "¿Desea cambiar este código de cuenta? *Y/N)";
          GET mconfirm PICTURE "Y"
   READ
   IF .NOT. mconfirm
      LOOP
   ENDIF
```

Figura 26-13.

Programa para cambiar el campo de clave de índice en los archivos relacionados.


```
   SELECT Customer
   SEEK moldacct
   REPLACE Account WITH mnewacct

   @ 10,0
   DO Center WITH 10,50 "W+", "Trabajando con el archivo Transacciones"
   SELECT Transact
   SEEK moldacct
   DO WHILE FOUND()
      REPLACE Account WITH mnewacct
      SEEK moldacct
   ENDDO

   @ 10,0
   DO Center WITH 10,60, "W+", "Trabajando con el archivo Pedidos"
   SELECT Order
   SEEK moldacct
   DO WHILE FOUND()
      REPLACE Account WITH mnewacct
      SEEK moldacct
   ENDDO

   @ 10,0
   DO Center WITH 10, 60, "W+", "Trabajando con el archivo Histórico de Tran-
   sacciones"
   SELECT Tranhist
   SEEK moldacct
   DO WHILE FOUND()
      REPLACE Account WITH mnewacct
      SEEK moldacct
   ENDDO

ENDDO

CLOSE DATABASES
DEACTIVATE WINDOW utility
RELEASE WINDOW utility
RETURN

* Fin del programa NWCHGACT.PRG
```

Figura 26-13.

Programa para cambiar el campo de clave de índice en los archivos relacionados *(continuación)*.

está asignado a ningún cliente. En ambos casos, el programa acepta un código de cuenta en blanco y permite que el usuario salga del programa dejando cualquiera de los códigos de cuenta vacíos.

Después de pedir confirmación al usuario, el programa modifica el campo AC-

COUNT del archivo de Clientes con una simple orden REPLACE. El cambio del campo ACCOUNT en los archivos relacionados es algo más complicado. Aunque puede utilizar una orden SEEK para encontrar el primer registro cuyo código de cuenta coincide con el original, no puede modificar todos los registros del grupo utilizando una orden REPLACE que incluya una cláusula WHILE, tal como ACCOUNT = MOLDACCT, ni puede utilizar el bucle DOWHILE o SCAN equivalente.

Cuando dBASE IV modifica el valor de un campo clave de índice, la posición de este registro en el índice se ajusta inmediatamente, pero el puntero de registro permanece posicionado en el registro modificado. Cuando dBASE IV lleva el puntero de registro al "siguiente" registro, este registro será en la mayoría de los casos uno que no es miembro del grupo original de registros relacionados, y la condición ACCOUNT = MOLDACCT no será cierta para este registro.

El programa NWCHGACT.PRG resuelve este problema buscando el primer registro coincidente en el archivo relacionado y luego utilizando un bucle DO WHILE gobernado por la condición FOUND( ) para procesar el grupo de registros que comparten el código de cuenta MOLDACCT. Dentro de este bucle, el programa sustituye el registro actual y luego repite la orden SEEK. Cuando todos los registros han sido reemplazados, la función FOUND se evalúa a .F. y el bucle DO WHILE deja de ejecutarse.

CAPITULO VEINTISIETE

Utilidades de mantenimiento del sistema

El Capítulo 26 describió cómo escribir programas de utilidad para realizar operaciones de mantenimiento de bases de datos, algunos de los cuales son específicos de una aplicación, y otros son necesarios en todos los sistemas, tales como los programas que empaquetan y reindexan todos los archivos de bases de datos utilizados en la aplicación. Este capítulo estudia otra clase de utilidades, aquellas que llevan a cabo actividades que son descritas como operaciones de mantenimiento del sistema, tales como la ejecución de órdenes del sistema operativo, ejecución de programas de utilidad para hacer copias de seguridad de las bases de datos o reponer los archivos de dichas copias, y configurar la aplicación para el hardware y la compañía del usuario.

Los programas de los 12 últimos capítulos hacían uso de variables de memoria para almacenar "opciones de ámbito global del sistema", que los usuarios debían tener disponibles para cualquier programa de la aplicación. El uso de variables de memoria en lugar de "codificar permanentemente" las cantidades que representan le permite construir aplicaciones que son mucho más adaptables a las necesidades de los usuarios o a las de otras organizaciones y aplicaciones. Este capítulo introduce algunas opciones de ámbito general adicionales y describe con detalle un programa de configuración que asigna valores por omisión a algunas de las variables de ámbito global y permite que los usuarios personalicen otras.

Este capítulo subraya la importancia que tiene para los usuarios y los programadores una aplicación y la programación de la misma independiente del hardware, y presenta una gran variedad de técnicas específicas que puede utilizar para evitar demasiados supuestos en los programas. Para los usuarios, esta aproximación ofrece una flexibilidad adicional ya que permite personalizar la aplicación sin programar sus necesidades de cambio o preferencias. Para el programador, ofrece mayor productividad ya que son necesarios pocos cambios para adaptar los programas escritos para una organización a las necesidades de otra.

Finalmente este capítulo describe una forma de hacer para que una aplicación sea más amigable para el usuario, añadiendo una selección de órdenes del sistema operativo a los menús, aprovechándose de la ventaja que ofrece dBASE IV al ejecutar cualquier orden o programa externo del DOS, permitiendo que los operadores que no están familiarizados con el sistema de las órdenes del MS-DOS lleven a cabo ciertas tareas de mantenimiento del sistema cruciales, tales como realizar copias de seguridad de las bases de datos o comprobar el espacio de disco disponibles. A su vez, puede proporcionar a los usuarios más experimentados una forma más conveniente de ejecutar órdenes del DOS u otros programas sin tener que salir de dBASE IV en el transcurso de una sesión de trabajo.

PERSONALIZACION DE LAS OPCIONES DE AMBITO GLOBAL DEL SISTEMA

El Capítulo 15 introdujo la idea del uso de variables de memoria para almacenar las opciones de ámbito global del sistema (cantidades que deberían estar disponibles para cualquier programa de una aplicación y que cambian con poca frecuencia). Hasta ahora, ha actualizado estas opciones desde el punto indicativo, restaurándose a partir del archivo de variables de memoria NW.MEM, inicializando las nuevas variables con órdenes STORE o sentencias de asignación, y almacenando el conjunto de nuevo en el disco. Cuando escribe una aplicación para que sea utilizada por otras personas, debería incluir un programa de configuración del sistema que permita que los usuarios modifiquen algunas de las opciones de ámbito global del sistema de modo que un simple cambio, tal como llevar la aplicación de la unidad C a la unidad D, o cambiar la dirección de la compañía, no requiera ningún conocimiento de dBASE IV. El programa de configuración también debe crear las variables de memoria restantes, que son desconocidas por los usuarios, para facilitar la recuperación desde un disco en el que el archivo de variables de memoria se haya perdido.

Alguna de las nuevas opciones de ámbito global añadidas en este capítulo, tales como variables que almacenan los colores de visualización y la unidad de disco que aloja la aplicación, se utilizan para hacer que la aplicación sea más independiente de su entorno. Otras permiten que los usuarios adapten el sistema a su organización y son únicas para cada aplicación (aunque algunas variables, tales como el nombre de la compañía, y la dirección pueden encontrarse en la mayoría de los sistemas). Si lo desea, puede escribir dos programas de configuración uno para las opciones específicas de la aplicación y otro para las opciones que se utilizarán sin modificaciones en todos sus sistemas de dBASE IV. Este último programa, que incluirá en todos los sistemas que escriba, podría llamar al otro de modo que la distinción sea transparente a los usuarios. El sistema National Widgets utiliza un único programa de configuración para personalizar todas las opciones de ámbito global, pero presenta dos grupos de opciones en dos pantallas de entrada independientes.

dBASE IV proporciona funciones incorporadas que le permiten determinar algunas características del hardware y del entorno del sistema operativo. Utilizando estas funciones, puede hacer que el programa sea independiente de su entorno sin tener que definir variables de memoria que almacenen valores específicos. Por ejemplo, puede utilizar la función ISCOLOR para determinar si el monitor en uso es de color/gráficos (esta función se describe con mayor detalle posteriormente en este apartado). Puede comprobar el valor de la función VERSION, que evalúa el nombre de la versión de dBASE IV, antes de ejecutar órdenes que no están soportadas en las versiones anteriores a dBASE IV. La función OS devuelve el nombre del sistema operativo como una cadena de caracteres, por ejemplo, "DOS 3.20". En una red, la función OS da por resultado el nombre de la versión DOS o OS/2 utilizada para arrancar la computadora antes de cargar el software de red, no el nombre de la red. Sin embargo, puede comprobar si dBASE IV se está ejecutando sobre una red evaluando el valor de la función NETWORK, que da por resultado .T. en una red de área local o .F. en una instalación monousuario.

Algunas opciones de ámbito global del sistema, tal como la unidad de disco que contiene los archivos de base de datos y las variables específicas de la aplicación deberían ser especificadas por el usuario. Además puede permitir que los usuarios sintonicen en entorno de trabajo para lograr sus preferencias personales seleccionando el estado de los parámetros del entorno. Por ejemplo, el programa de configuración del sistema National Widgets permite que los usuarios especifiquen si los parámetros BELL y CONFIRM deben estar en ON o en OFF y seleccionen el formato de visualización de las fechas.

Para otras opciones, tales como los colores de visualización o los atributos monocromáticos, la elección no es tan clara Almacenando los códigos de color en variables de memoria, puede personalizar la comunicación con el usuario en lugar de establecer combinaciones de color que funcionen en cualquier monitor, pero que no coincida con los colores utilizados en las ventanas o menús visualizados simultáneamente. Un ejemplo de esto es el blanco brillante sobre negro utilizado en el procedimiento SCRNHEAD, que visualiza una cabecera en la parte superior de la pantalla para describir el programa que se encuentra en ejecución actualmente. Muchos de los procedimientos del sistema National Widgets, incluyendo aquellos que centran el texto y visualizan mensajes, han sido diseñados para acomodar las especificaciones del color aceptando los colores de visualización como un parámetro. Pasando al procedimiento una variable de memoria en lugar de una cadena de caracteres explícita, puede hacer que el procedimiento que llama sea tan general como los procedimientos en sí mismos. Si posteriormente cambia de forma de pensar sobre los colores, puede modificar la variable de las opciones de ámbito global relevante.

Puesto que las opciones, tales como los colores de visualización, juegan un papel importante en la "apariencia" de la aplicación, los usuarios tienen la oportunidad de personalizarlas. Sin embargo, incluso si han trabajado con dBASE IV en el punto indicativo o a través del sistema de menús, no puede suponer que los usuarios comprenden todos los elementos que controlan los colores de la pantalla, que puede personalizar con órdenes SET COLOR OF, y puede que prefiera mantener el control del color de modo que los menús, cajas de diálogo, y mensajes de aviso tengan una apariencia consistente en todas sus aplicaciones.

Además, escribir un programa para seleccionar los colores de fondo y primer plano de todos los objetos de la pantalla no es una hazaña cualquiera. Para permitir que los usuarios seleccionen los colores sin saber las abreviaturas del color de dBASE IV, el programa de configuración tendría que presentar una lista de todas las selecciones

posibles para los colores de fondo y primer plano (o atributos monocromáticos), preferiblemente acompañados por ejemplos de los colores disponibles. En efecto, tendría que emular el menú **Pantalla (Display)** disponible en el sistema de menús de dBASE IV (por ejemplo, en el editor de diseño de pantalla y en el generador de aplicaciones). El programa de configuración del sistema National Widgets que se describe en el siguiente apartado, pone de manifiesto una técnica para hacer esto y permite que los usuarios seleccionen los colores de visualización de la pantalla principal (los colores especificados en la orden SET COLOR OF NORMAL) y los de los datos introducidos (los colores establecicos con SET COLOR OF FIELDS) si dispone de un monitor de color. En sus propias aplicaciones, puede que desee ofrecer a los usuarios otras selecciones adicionales, o puede que decida seleccionar todos los colores de visualización por usted mismo y modificarlos si los usuarios requieren modificaciones posteriormente.

Una versión revisada del programa principal de la aplicación, que llama al programa de configuración y tiene en cuenta todas las variables de opciones de ámbito global del sistema, aparece en la Figura 27-1. NW2.PRG llama al programa de configuración automáticamente si el archivo de variables de memoria NW.MEM no se

```
* NW2.PRG
* PROGRAMA DEL MENU PRINCIPAL DEL SISTEMA NATIONAL WIDGETS
* ULTIMA REVISION: 07/18/89 M. LISKIN

* Reinicia el entorno de trabajo
CLEAR ALL
CLOSE ALL
SET CONSOLE ON
SET PRINT OFF
SET DEVICE TO SCREEN
SET TALK OFF

* Configura el entorno de trabajo

* Declare las variables de ámbito global del sistema públicas
* Parámetros del entorno
PUBLIC msbell, msconfirm, msdate, mspdriver
* Colores de visualización
PUBLIC msnormal, mshigh, msmessages, mstitles, msbox, msinfo, msfields,;
   mswarning
* Mensajes estándar
PUBLIC mspresskey, msfirst, mslast, msonfile, msaskdel, msdeleted,;
   msrecover, msnotfound
* Opciones dependientes de la aplicación
PUBLIC mscompany, msadd1, msadd2, msadd3, msstate, msareacode, mstaxrate,;
   msinvoice, msdrive, msprefix
```

Figura 27-1.

Una nueva versión del programa de arranque de la aplicación.


```
* Abre el archivo de procedimientos principal del sistema
SET PROCEDURE TO NWproc

* Abre el archivo de ayuda del sistema
ON KEY LABEL F1 DO NWhelp2 WITH PROGRAM()

* Restaura las variables de memoria y las macros
IF FILE("NW.MEM")                       && Si las variables de ámbito
                                        global existen,
   RESTORE FROM NW ADDITIVE             &&    restaura el archivo
ELSE                                    && En cualquier otro caso, ejecu-
                                        ta el programa de configuración
DO NWsetup WITH.T.                      &&    señal = define las varia-
                                        bles como nuevas
ENDIF
RESTORE MACROS FROM NWprogrm

* Guarda el estado del entorno en variables de memoria
meautosave = SET("AUTOSAVE")
mebell     = SET("BELL")
mecentury  = SET("CENTURY")
meclock    = SET("CLOCK")
meconfirm  = SET("CONFIRM")
medeleted  = SET("DELETED")
medelimit  = SET("DELIMITERS")
meexact    = SET("EXACT")
meintens   = SET("INTENSITY")
menear     = SET("NEAR")
mesafety   = SET("SAFETY")
mescorebd  = SET("SCOREBOARD")
mestatus   = SET("STATUS")

* Personaliza el entorno con órdenes SET
SET AUTOSAVE ON
SET BELL &msbell.
SET BORDER TO SINGLE
SET CENTURY OFF
SET CLOCK OFF
SET CONFIRM &msconfirm.
SET DATE &msdate.
SET DELETED ON
SET DELIMITERS OFF
SET ESCAPE ON
SET EXACT OFF
SET INTENSITY ON
SET NEAR OFF
SET SAFETY OFF
```

Figura 27-1.

Una nueva versión del programa de arranque de la aplicación *(continuación)*.

```
SET SCOREBOARD ON
SET STATUS OFF

SET COLOR OF NORMAL TO &msnormal.
SET COLOR OF HIGHLIGHT TO &mshigh.
SET COLOR OF MESSAGES TO &msmessages.
SET COLOR OF TITLES TO &mstitles.
SET COLOR OF BOX TO &msbox.
SET COLOR OF INFORMATION TO &msinfo.
SET COLOR OF FIELDS TO &msfields.

* Visualiza la pantalla de presentación
DO Signon

* Define los menús y las ventanas
DO Menudef

* Visualiza la pantalla del menú principal
DO Scrnhead WITH msbox,;
   "Sistema de seguimiento de Clientes y Pedidos del sistema National Widgets"
ACTIVATE POPUP mainmenu

* Restaura el estado anterior del entorno de trabajo
SET AUTOSAVE &meautosave.
SET BELL &mebell.
SET CENTURY &mecentury.
SET CLOCK &meclock.
SET CONFIRM &meconfirm.
SET DELETED &medeleted.
SET DELIMITERS &medelimit.
SET EXACT &meexact.
SET INTENSITY &meintens.
SET NEAR &menear.
SET SAFERY &mesafety.
SET SCOREBOARD &mescorebd.
SET STATUS &mestatus.

CLEAR ALL
CLOSE ALL
CLEAR
RETURN

*************************************************************************

* Visualiza la pantalla de presentación
PROCEDURE Signon
```

Figura 27-1.

Una nueva versión del programa de arranque de la aplicación *(continuación)*.


```
CLEAR
@ 5,10 TO 16,70 DOUBLE
DO Center WITH  8,80 "W+/N", "SISTEMA DE SEGUIMIENTO DE CLIENTES Y PEDIDOS"
DO Center WITH 10,80 "", "National Widgets, Inc."
DO Center WITH 14,80 "", "Copyright (C) 1989 Miriam Liskin. "+;
   "Todos los derechos reservados."
DO Center WITH 23,80, "", "Pulse cualquier tecla para visualizar el menú principal"
mkey = INKEY(10)                              && Hace una pausa de 10 segundos
CLEAR
RETURN

* Fin del procedimiento Signon

* Define las ventanas y los menús del sistema
PROCEDURE Menudef

* Ventana de pantalla completa
DEFINE WINDOW fullscrn FROM 0,0 TO 24,79 NONE
* Ventana de pantalla completa para ejecutar las acciones del menú
DEFINE WINDOW menuscrn FROM 0,0 TO 24,79 NONE
* Ventana para usar la orden BROWSE con la línea de estado activada
DEFINE WINDOW browscrn FROM 1,0 TO 21,79 NONE
* Ventana para monitorizar los procesos secuenciales y los mensajes de error
DEFINE WINDOW monitor FROM 10,10 TO 18,70 DOUBLE
* Ventana para solicitar las selecciones y entradas del usuario
DEFINE WINDOW helpscrn FROM 3,5 TO 21,75

ACTIVATE WINDOW fullscrn

* Pantalla del menú principal
DEFINE POPUP mainmenu FROM 5,26 TO 20,54;
   MESSAGE "Seleccione una opción, destáquela y pulse <ENTER>"+;
   "o pulse la letra inicial"
DEFINE BAR  1 OF mainmenu PROMPT      "Menú Principal"      SKIP
DEFINE BAR  2 OF mainmenu PROMPT "------------------------" SKIP
DEFINE BAR  3 OF mainmenu PROMPT " Clientes: entrada";
   MESSAGE "Ver, editar, añadir y borrar clientes"
DEFINE BAR  4 OF mainmenu PROMPT "Inventario: entrada";
   MESSAGE "Ver, editar, añadir y borrar artículos del inventario"
DEFINE BAR  5 OF mainmenu PROMPT "Pedidos entrada";
   MESSAGE "Introducir pedidos y pagos por anticipado"
DEFINE BAR  6 OF mainmenu PROMPT "Transacciones entrada";
   MESSAGE "Ver, editar, añadir y borrar transacciones financieras"
DEFINE BAR  7 OF mainmenu PROMPT "------------------------" SKIP
DEFINE BAR  8 OF mainmenu PROMPT "Etiquetas y cartas";
```

Figura 27-1.

Una nueva versión del programa de arranque de la aplicación *(continuación)*.

```
   MESSAGE "Visualiza el menú de las opciones de cartas pesonalizadas y eti-
           quetas para correspondencia"
DEFINE BAR  9 OF mainmenu PROMPT "Informes";
   MESSAGE "Visualiza el menú de informes"
DEFINE BAR 10 OF mainmenu PROMPT "-------------------------" KIP
DEFINE BAR 11 OF mainmenu PROMPT "Utilidades";
   MESSAGE "Visualiza el menú de utilidades"
DEFINE BAR 12 OF mainmenu PROMPT "-------------------------" SKIP
DEFINE BAR 13 OF mainmenu PROMPT "Salir a dBASE IV"
   MESSAGE "Sale de esta aplicación al punto indicativo"
DEFINE BAR 14 OF mainmenu PROMPT "Salir al Sistema Operativo";
   MESSAGE "Sale de la aplicación y de dBASE IV"
ON SELECTION POPUP mainmenu DO NWmmenu

* Menú de etiquetas y cartas
DEFINE POPUP lblmenu FROM 7,26 TO 15,53;
   MESSAGE "Seleccione una opción, destáquela y pulse <ENTER>"+,
   "o pulse la letra inicial"
DEFINE BAR  1 OF lblmenu PROMPT "Etiquetas y cartas"    SKIP
DEFINE BAR  2 OF lblmenu PROMPT "---------------------" SKIP
DEFINE BAR  3 OF lblmenu PROMPT "Etiquetas para correspondencia"
DEFINE BAR  4 OF lblmenu PROMPT "Cartas personalizadas"
DEFINE BAR  5 OF lblmenu PROMPT "Sobres"
DEFINE BAR  6 OF lblmenu PROMPT "Sobres de tarjetas de felicitación"
DEFINE BAR  7 OF lblmenu PROMPT "Otras etiquetas"
ON SELECTION POPUP lblmenu DO NWrmenu

* Menú informes
DEFINE POPUP rptmenu FROM 1,21 TO 22,58;
   MESSAGE "Seleccione una opción, destáquela y pulse <ENTER> "+,
   "o pulse la letra inicial"
DEFINE BAR  1 OF rptmenu PROMPT "   Menú informes" SKIP
DEFINE BAR  2 OF rptmenu PROMPT "---------------------" SKIP
DEFINE BAR  3 OF rptmenu PROMPT "Lista de referencia de clientes"
DEFINE BAR  4 OF rptmenu PROMPT "Tarjetas Rodolex de clientes"
DEFINE BAR  5 OF rptmenu PROMPT "Perfil del equipo del cliente"
DEFINE BAR  6 OF rptmenu PROMPT "Perfil de tipo de producto del cliente"
DEFINE BAR  7 OF rptmenu PROMPT "Facturas"
DEFINE BAR  8 OF rptmenu PROMPT "Estado de cuentas mensual"
DEFINE BAR  9 OF rptmenu PROMPT "---------------------" SKIP
DEFINE BAR 10 OF rptmenu PROMPT "Lista de referencia del Inventario"
DEFINE BAR 11 OF rptmenu PROMPT "Lista de precios"
DEFINE BAR 12 OF rptmenu PROMPT "Resumen inventario por tipo/identificación"
DEFINE BAR 13 OF rptmenu PROMPT "----------------------" SKIP
DEFINE BAR 14 OF rptmenu PROMPT "Lista de referencia de pedidos"
DEFINE BAR 15 OF rptmenu PROMPT "Resumen por tipo/número identificación"
```

Figura 27-1.

Una nueva versión del programa de arranque de la aplicación *(continuación)*.


```
DEFINE BAR 16 OF rptmenu PROMPT "Lista de referencia de transacciones"
DEFINE BAR 17 OF rptmenu PROMPT "Registro de facturas"
DEFINE BAR 18 OF rptmenu PROMPT "Lista ingresos en efectivo"
DEFINE BAR 19 OF rptmenu PROMPT "Informe de atrasos"
DEFINE BAR 20 OF rptmenu PROMPT "Resumen de contabilidad anual"
ON SELECTION POPUP rptmenu DO NWmenu

RETURN
* Fin del procedimiento Menudef

* Fin del programa NW2.PRG
```

Figura 27-1.

Una nueva versión del programa de arranque de la aplicación *(continuación)*.

encuentra en el disco, como podría suceder si el usuario arranca el sistema por primera vez, o si el archivo de memoria se ha perdido o dañado en el disco. El usuario puede ejecutar también el programa de configuración en cualquier instante eligiendo la opción en el menú de utilidades que se muestra en la Figura 27-2. El programa que define el menú de utilidades y ejecuta las opciones se lista en la Figura 27-3. La estructura general del menú de utilidades es algo diferente del programa que ejecuta

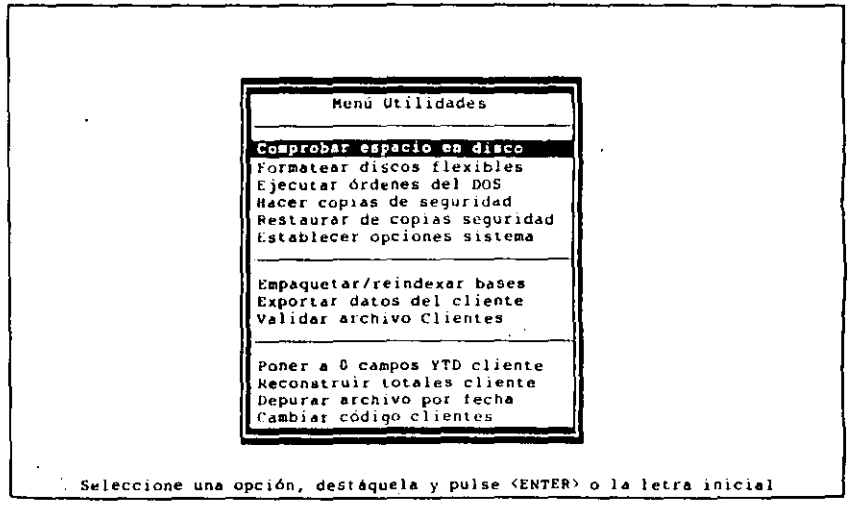

Figura 27-2.

El menú Utilidades.

```
* NWUMENU.PRG
* ACCIONES DEL MENU UTILIDADES
* ULTIMA REVISION: 07/18/89 M. LISKIN

DO Menudef

ACTIVATE WINDOW utilmenu
ACTIVATE POPUP utilmenu

DEACTIVATE WINDOW utilmenu
RELEASE WINDOW utilmenu
RELEASE POPUP utilmenu
RETURN

*************************************************************************

Define las ventanas y el menú
PROCEDURE Menudef

* Ventana para visualizar el menú
DEFINE WINDOW utilmenu FROM 0,0 TO 24,79 NONE

* Opciones del menú Utilidades

DEFINE POPUP utilmenu FROM 4,21 TO 21,55;
   MESSAGE "Seleccione una opción, destáquela y pulse <ENTER> "+,
   "o pulse la letra inicial"
DEFINE BAR  1 OF utilmenu PROMPT "   Menú utilidades" SKIP
DEFINE BAR  2 OF utilmenu PROMPT "---------------------" SKIP
DEFINE BAR  3 OF utilmenu PROMPT "Comprobar espacio en disco"
DEFINE BAR  4 OF utilmenu PROMPT "Formatear discos flexibles"
DEFINE BAR  5 OF utilmenu PROMPT "Ejecutar órdenes del DOS"
DEFINE BAR  6 OF utilmenu PROMPT "Copias seguridad bases de datos"
DEFINE BAR  7 OF utilmenu PROMPT "Restaurar copias de seguridad"
DEFINE BAR  8 OF utilmenu PROMPT "Configurar opciones del sistema"
DEFINE BAR  9 OF utilmenu PROMPT "---------------------" SKIP
DEFINE BAR 10 OF utilmenu PROMPT "Empaquetar y/o Reindexar bases de datos"
DEFINE BAR 11 OF utilmenu PROMPT "Exportar datos de clientes"
DEFINE BAR 12 OF utilmenu PROMPT "Validar el archivo Clientes"
DEFINE BAR 13 OF utilmenu PROMPT "---------------------" SKIP
DEFINE BAR 14 OF utilmenu PROMPT "Poner a cero campos YTD de clientes"
DEFINE BAR 15 OF utilmenu PROMPT "Reconstruir total de los clientes"
DEFINE BAR 16 OF utilmenu PROMPT "Depurar archivos por fecha"
DEFINE BAR 17 OF utilmenu PROMPT "Cambiar el código de cuenta de clientes"
ON SELECTION POPUP utilmenu DO Utilmenu

RETURN
```

Figura 27-3.

Programa que ejecuta las opciones del menú Utilidades.


```
* Acciones del menú Utilidades
PROCEDURE Utilmenu

DO CASE
   CASE BAR() = 3                 && Comprueba el espacio en disco
      CLEAR
      RUN CHKDSK
   CASE BAR() = 4                 && Formatea los discos
      RUN FORMAT A:
   CASE BAR() = 5                 && Ejecuta órdenes del DOS
      DO NWrundos
   CASE BAR() = 6                 && Copias de seguridad de las bases de datos
      USE MW.CAT ALIAS Filelist   && Abre el catálogo como una base de datos
      DO NWbackup
   CASE BAR() = 7                 && Restaura las bases de datos
      DO NWRestor
   CASE BAR() = 8                 && Configura las opciones del sistema
      DO NWsetup WITH .F.
   CASE BAR() = 10                && Empaqueta y/o reindexa las bases de datos
      USE NW.CAT ALIAS Filelist   && Abre el catálogo como una base de datos
      DO NWindex4
   CASE BAR() = 11                && Exporta los datos del cliente
      DO NWcopy
   CASE BAR() = 12                && Valida el archivo de clientes
      DO NWcvalid
   CASE BAR() = 14                && Pone a cero los campos YTD de los clientes
      DO NWczero
   CASE BAR() = 15                && Reconstruye el total de los clientes
      DO NWtotal
   CASE BAR() = 16                && Depura los archivos por la fecha
      DO NWpurge
   CASE BAR() = 17                && Modifica el código de cuenta del cliente
      DO NWchgact
ENDCASE

RETURN

* Fin del procedimiento Utilmenu

* Fin del programa NWUMENU.PRG
```

Figura 27-3.

Programa que ejecuta las opciones del menú Utilidades *(continuación)*.

las opciones del menú principal puesto que el menú de utilidades se utiliza con menor frecuencia que el menú principal y el menú de informes. El menú y las ventanas se definen en el mismo programa que ejecuta las opciones de menú para que no ocupen memoria cuando no estén en uso. Muchas opciones deberían serle familiares del Capítulo 26, y las restantes se describen en este capítulo.

Si NW2.PRG llama al programa de configuración automáticamente debe crear todas las variables rqueridas desde el principio utilizando los valores por omisión establecidos por el programador. Si el usuario ejecuta el programa de configuración intencionalmente, no debería reinicializar o alterar ninguna de las variables. Para hacer distinción entre estas dos posibilidades, el programa de configuración acepta un parámetro del programa que llama una variable lógica llamada MNEWFILE. El programa de arranque pasa el valor .T. a MNEWFILE para indicar que el programa de configuración debe crear un nuevo archivo de variables de memoria. Cuando el usuario selecciona la opción **Configuración del sistema** del menú Utilidades, el programa NWUMENU.PRG pasa el valor .F. a MNEWFILE en la orden DO que llama al programa de configuración.

Al igual que la primera versión del programa de arranque del sistema, NW2.PRG declara las variables de ámbito global como públicas antes de reponer el archivo de variables de memoria o llamar al programa de configuración que las crea. dBASE IV no guarda en el archivo de variables de memoria la declaración de la misma como pública, y por tanto después de declarar la variable como pública, debe incluir la palabra clave ADDITIVE en la orden RESTORE para evitar que dBASE IV elimine las variables creadas con la declaración PUBLIC de la memoria. El programa NW2.PRG utiliza una serie de órdenes PUBLIC para declarar todas las variables requeridas como públicas en grupos lógicos que reflejan su uso en la aplicación de National Widgets.

No es necesario declarar las variables de opciones de ámbito global de National Widgets como públicas, ya que han sido definidas en el programa principal de la aplicación. Todos los programas o procedimientos de la aplicación son llamados directamente o indirectamente por este programa y en consecuencia tienen acceso a las variables creadas por él. Sin embargo, muchos programadores prefieren usar procedimientos o programas independientes para inicializar las variables utilizadas en toda la aplicación, y en este caso *debe* declarar las variables como públicas para hacerlas disponibles a todos los programas y procedimientos del sistema.

El programa de arranque utiliza la función FILE para comprobar la presencia del archivo de variables de memoria NW.MEM. Si existe este archivo, ejecuta una orden RESTORE para introducir las variables en la memoria; en caso contrario, llama al programa de configuración, pasa el valor .T. al parámetro MNEWFILE para indicar que las variables de ámbito global deben crearse desde el principio.

Un programa para configurar las opciones de ámbito global

El programa de configuración del sistema National Widgets, NWSETUP.PRG, se lista en la Figura 27-4. Este abre el archivo Códigos para validar las entradas del usuario para el estado y el código de área de los nuevos clientes; define una variable de memoria llamada MCOLORS, que se utiliza en los procedimientos para que los usuarios seleccionen los colores de visualización; y luego llama al procedimiento MENUDEF para definir las listas de selección que visualizan los formatos de fecha disponibles, controladores de impresora, y colores de visualización. Si MNEWFILE es .T., el programa de configuración ha sido llamado automáticamente por NW2.PRG, por-


```
* NWSETUP.PRG
* PROGRAMA PARA CONFIGURAR LAS VARIABLES DE OPCIONES DE AMBITO GLOBAL
* ULTIMA REVISION: 08/16/89 M. LISKIN

PARAMETER mnewfile

* Abre el archivo de códigos para validar el estado y el código de área
USE NWcodes ORDER Codetype ALIAS Codes

* Códigos y nombres de color (usados por los procedimientos Foreground y Background)
mcolors = "/Black /N /Blue /B /Green / G /Cyan /BG" +;
          "/Red /R /Magenta /RB /Brown /GR /White /W" +;
          "/Gray /N+ /Lt Blue /B+ /Lt Grn /G+ /Lt Cyan /BG+" +;
          "Lt Red/R+ /Lt Mag /RB+ /Yellow /GR+ /Brt Wht /W+ "

DO Menudef                                  && Define las listas de selección

IF mnewfile
   DO Defaults                              && Valores por omisión en blanco
ENDIF
DO Colors                                   && Establece los colores de la
                                                  visualización
DO Stdmsgs                                  && Mensajes estándar del sistema

DO Scrnhead WITH msbox, "Set Up System Options"
DEFINE WINDOW setup FROM 5,0 TO 23,79 DOUBLE
ACTIVATE WINDOW setup

* Opciones independientes de la aplicación

* Parámetros del entorno
* Inicia las variables temporales según las opciones de ámbito global
mbell = IIF(msbell = "OFF", .T., .F.)
mconfirm = IIF(msconfirm = "ON", .T., .F.)
* Capta las variables temporales
@  1,5 SAY "¿Desea quitar el sonido (el zumbador)? (Y/N)";
       GET mbell PICTURE "Y"
@  2,5 SAY "¿Desea pulsar <ENTER> después de cada entrada? (Y/N)";
       GET mconfirm PICTURE "Y"

READ
* Establece las opciones de ámbito global según las variables temporales
msbell = IIF(mbell, "OFF", "ON")
msconfirm = IIF(mconfirm, "ON", "OFF")
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema.

```
* Formato de visualización de las fechas
mdate = "MM/DD/YY"
msdate = "AMERICAN"
@  4,5 SAY "Elija el formato de visualización de las fechas:"
ACTIVATE POPUP dateform
@  4,40 SAY mdate

* Selecciona el color del texto y del campo en un sistema de color
IF ISCOLOR()
   DO Colorbar                              && Visualiza el color de las líneas
   mforegrnd = "W"                          && Color del primer plano
   mbackgrnd = "B"                          && Color del fondo
   @  6, 5 SAY "Elija los colores del texto:"
   ACTIVATE POPUP foregrnd                  && Selección del color de primer plano
   ACTIVATE POPUP backgrnd                  && Selección del color de fondo
   msnormal = mforegrnd + "/" + mbackgrnd   && Establece los colores del texto
   @  6,40 SAY "Text colors" COLOR &msnormal && Visualiza los colores del texto
   mforegrnd = "N"                          && Color del primer plano
   mbackgrnd = "BG"                         && Color del fondo
   @  7, 5 SAY "Elija colores para los datos:"
   ACTIVATE POPUP foregrnd                  && Selección del color de primer plano
   ACTIVATE POPUP backgrnd                  && Selección del color de fondo
   msfields = mforegrnd + "/" + mbackgrnd   && Establece el color de los campos
   @  7,40 SAY "Data colors" COLOR &msfields.    && Visualiza el color de los campos
ENDIF

* Establece todos los colores de visualización según las variables de ámbito
global del sistema
SET COLOR OF NORMAL TO &msnormal.
SET COLOR OF HIGHLIGHT TO &mshigh.
SET COLOR OF MESSAGES TO &msmessages.
SET COLOR OF TITLES TO &mstitles.
SET COLOR OF BOX TO &msbox.
SET COLOR OF INFORMATION TO &msinfo.
SET COLOR OF FIELDS TO &msfields.

* Impresora por omisión
@  9, 5 SAY "Elija la impresora por omisión:"
ACTIVATE POPUP divers
@  9,40 SAY mspdriver
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema *(continuación)*.


```
* Unidad de disco de la aplicación
@ 11, 5 SAY "Introduzca la unidad de disco en la que se encuentra el sistema ";
        GET msdrive FUNCTION "A!"
READ

* Opciones específicas de la aplicación
CLEAR
@  1, 5 SAY "Introduzca el nombre y la dirección de la compañía:"
@  2,10 SAY "Compañía" GET mscompany
@  3,10 SAY "Dirección" GET msadd1
@  4,18 GET msadd2
@  5,18 GET msadd3

@  7, 5 SAY "Introduzca el estado por omisión de los nuevos clientes ";
        GET mstate FUNCTION "A!" VALID SEEK("S"+State, "Codes");
        ERROR "No es una abreviatura válida"
@  9, 5 SAY "Introduzca el código de área por omisión de los nuevos clientes ";
        GET msareacode PICTURE "999" VALID SEEK("A"+Areacode, "Codes");
        ERROR "No es un código de área válido"
@ 11, 5 SAY "Introduzca el impuesto sobre las ventas por omisión en % (p.e.,
           6.50)";
        GET mstaxrate PICTURE "99.99"
@ 13, 5 SAY "Introduzca el siguiente número de factu : disponible ";
        GET msinvoice PICTURE "99999"
READ

SAVE ALL LIKE MS* TO NW
DEACTIVATE WINDOW setup
RELEASE WINDOW setup
RELEASE POPUP dateform, drivers, foregrnd, backgrnd
CLEAR
RETURN

*************************************************************************

* Define las listas de selección
PROCEDURE Menudef

* Lista de selección de formatos de fecha DEFINE POPUP dateform FROM 3.60;
   MESSAGE "Seleccione un formato de fecha, destáquelo y pulse <ENTER>"
DEFINE BAR 1 OF dateform PROMPT "MM/DD/YY"
DEFINE BAR 2 OF dateform PROMPT "MM-DD-YY"
DEFINE BAR 3 OF dateform PROMPT "DD/MM/YY"
DEFINE BAR 4 OF dateform PROMPT "DD.MM.YY"
DEFINE BAR 5 OF dateform PROMPT "YY/MM/DD"
DEFINE BAR 6 OF dateform PROMPT "YY.MM.DD"
ON SELECTION POPUP dateform DO Dateform
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema *(continuación)*.

```
* Lista de selección de las impresoras
DEFINE POPUP drivers FROM 1,62 TO 11,75 PROMPT FILES LIKE *.PR2;
   MESSAGE "Seleccione una impresora, destáquela y pulse <ENTER>"
ON SELECTION POPUP drivers DO Drivers

* Lista de selección de los colores de primer plano
DEFINE POPUP foregrnd FROM 1,64 TO 11,75;
   MESSAGE "Seleccione un color para el primer plano, destáquelo y pulse <EN-
TER>"
DEFINE BAR  1 OF foregrnd PROMPT "Negro"
DEFINE BAR  2 OF foregrnd PROMPT "Azul"
DEFINE BAR  3 OF foregrnd PROMPT "Verde"
DEFINE BAR  4 OF foregrnd PROMPT "Cian"
DEFINE BAR  5 OF foregrnd PROMPT "Rojo"
DEFINE BAR  6 OF foregrnd PROMPT "Magenta"
DEFINE BAR  7 OF foregrnd PROMPT "Marrón"
DEFINE BAR  8 OF foregrnd PROMPT "Blanco"
DEFINE BAR  9 OF foregrnd PROMPT "Gris"
DEFINE BAR 10 OF foregrnd PROMPT "Azul claro"
DEFINE BAR 11 OF foregrnd PROMPT "Verde claro"
DEFINE BAR 12 OF foregrnd PROMPT "Cian claro"
DEFINE BAR 13 OF foregrnd PROMPT "Rosa"
DEFINE BAR 14 OF foregrnd PROMPT "Magenta claro"
DEFINE BAR 15 OF foregrnd PROMPT "Amarillo"
DEFINE BAR 16 OF foregrnd PROMPT "Blanco brillante"
ON SELECTION POPUP foregrnd DO Foregrnd

* Lista de selección de los colores de fondo
DEFINE POPUP backgrnd FROM 1,64 TO 10,75;
   MESSAGE "Elija un color para el fondo, destáquelo y pulse <ENTER>"
DEFINE BAR 1 OF backgrnd PROMPT "Negro"
DEFINE BAR 2 OF backgrnd PROMPT "Azul"
DEFINE BAR 3 OF backgrnd PROMPT "Verde"
DEFINE BAR 4 OF backgrnd PROMPT "Cian"
DEFINE BAR 5 OF backgrnd PROMPT "Rojo"
DEFINE BAR 6 OF backgrnd PROMPT "Magenta"
DEFINE BAR 7 OF backgrnd PROMPT "Marrón"
DEFINE BAR 8 OF backgrnd PROMPT "Blanco"
ON SELECTION POPUP backgrnd DO Backgrnd

RETURN
* Fin del procedimiento Menudef

* Establece los valores omisión/blanco de algunas opciones
PROCEDURE Defaults
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema *(continuación)*.


```
msbell = "ON"                          && Parámetro BELL
msconfirm = "OFF"                      && Parámetro CONFIRM
msdate = "AMERICAN"                    && Parámmetro DATE
mspdriver = "GENERIC.PR2"              && Impresora
msdrive = "C"                          && Unidad de disco
msprefix = "NW"                        && Prefijo del nombre del archivo
                                          del sistema

store SPACE(30 TO mscompany, msadd1, msadd2, msadd3   && Compañía y dirección
msstate = " "                          && Estado por omisión
mseareacode = " "                      && Código del área por omisión
mstaxrate = 0.00                       && Impuesto sobre las ventas por
                                          omisión
msinvoice = 10000                      && Siguiente número de factura

RETURN
* Fin del procedimiento Defaults

* Establece los atributos de visualización colores/monocromáticos
PROCEDURE Colors

IF ISCOLOR()
   msnormal = "W/B"                    && blanco sobre azul
   mshigh = "GR+/R"                    && amarillo sobre rojo
   msmessages = "W/N"                  && blanco sobre negro
   mstitles = "GR+/B"                  && amarillo sobre azul"
   msbox = "W/R"                       && blanco sobre rojo
   msinfo = "G/N"                      && verde sobre negro
   msfields = "N/BG"                   && negro sobre azul claro
   mwarning = "R*/N"                   && rojo parpadeante sobre negro
ELSE
   msnormal = "W/N"                    && claro sobre oscuro
   mshight = "I"                       && inverso (oscuro sobre claro)
   msmessages = "W/N"                  && claro sobre oscuro
   mstitles = "W+"                     && brillante sobre oscuro
   msbox = "I"                         && inverso
   msinfo = "I"                        && inverso
   msfields = "U"                      && subrayado
   mwarning = "W+*/N"                  && brillante parpadeante sobre
                                          oscuro
ENDIF

RETURN
* Fin del procedimiento Colors

* Mensajes estándar de la aplicación
PROCEDURE Stdmsgs
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema *(continuación)*.

```
mspresskey = "-- Pulse cualquier tecla para continuar"
msfirst = "Esta es la primera"
mslast = "Esta es la última"
msonfile = "ya existe en el archivo"
msaskdel = "¿Está seguro que desea marcar este?"
msdeleted = "ha sido marcado"
msrecover = "ha sido desmarcado"
msnotfound = "no se encuentra"
msnotavail = "Esta opción no está disponible"

RETURN
* Fin del procedimiento Stdmsgs

* Procesa la fecha seleccionada de la lista
PROCEDURE Dateform

mdate = PROMPT()                          && Formato de visualización

* Asigna la palabra clave a la orden SET DATE
DO CASE
   CASE BAR() = 1
      msdate = "AMERICAN"
   CASE BAR() = 2
      msdate = "USA"
   CASE BAR() = 3
      msdate = "BRITISH"
   CASE BAR() = 4
      msdate = "GERMAN"
   CASE BAR() = 5
      msdate = "JAPAN"
   CASE BAR() = 6
      msdate = "ANSI"
ENDCASE

DEACTIVATE POPUP
RETURN
* Fin del procedimiento Dateform

* Procesa la impresora elegida de la lista de selección
PROCEDURE Drivers
mspdriver = PROMPT()
DEACTIVATE POPUP
RETURN
* Fin del procedimiento Drivers

* Visualiza una muestra de color de las líneas
PROCEDURE Colorbar
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema *(continuación)*.


```
@ 13, 1 SAY "Primer plano:"
@ 13,13 SAY "Negro   " COLOR N/W
@ 13,21 SAY "Azul    " COLOR B/N
@ 13,29 SAY "Verde   " COLOR G/N
@ 13,37 SAY "Cian    " COLOR BG/N
@ 13,45 SAY "Rojo    " COLOR R/N
@ 13,53 SAY "Magenta " COLOR RB/N
@ 13,61 SAY "Marrón  " COLOR GR/N
@ 13,69 SAY "Blanco  " COLOR W/N
@ 14,13 SAY "Gris    " COLOR N+/W
@ 14,21 SAY "Azul Cl " COLOR B+/N
@ 14,29 SAY "Verde Cl" COLOR G+/N
@ 14,37 SAY "Cian Cl " COLOR BG+/N
@ 14,45 SAY "Rojo Cl " COLOR R+/N
@ 14,53 SAY "Mag Cl  " COLOR RB+/N
@ 14,61 SAY "Amarillo" COLOR GR+/N
@ 14,69 SAY "Blan Br " COLOR W+/N
@ 16, 1 SAY "Fondo"
@ 16,13 SAY "Negro   " COLOR W/N
@ 16,21 SAY "Azul    " COLOR N/B
@ 16,29 SAY "Verde   " COLOR N/G
@ 16,37 SAY "Cian    " COLOR N/BG
@ 16,45 SAY "Rojo    " COLOR N/R
@ 16,53 SAY "Magenta " COLOR N/RB
@ 16,61 SAY "Marrón  " COLOR N/GR
@ 16,69 SAY "Blanco  " COLOR N/W
RETURN
* Fin del procedimiento Colorbar

* Procesa el color de primer plano elegido de la lista de selección
* Extrae el código de color de MCOLORS buscando el nombre del color
PROCEDURE Foregrnd
mcolorname = LEFT(PROMPT() + SPACE(7),7)  && Añade espacios al nombre del color
mforegrnd = TRIM(SUBSTR(mcolors, AT("/"+mcolorname+"/", mcolors)+ 9,3))
DEACTIVATE POPUP
RETURN
* Fin del procedimiento Foregrnd

* Procesa el color de fondo elegido de la lista de selección
* Extrae el código de color de MCOLORS buscando el nombre del color
PROCEDURE Backgrnd
mcolorname = LEFT(PROMPT() + SPACE(7),7)  && Añade espacios al nombre del color
mbackgrnd = TRIM(SUBSTR(mcolors, AT("/"+mcolorname+"/", mcolors)+ 9,3))
DEACTIVATE POPUP
RETURN
* Fin del procedimiento Backgrnd

* Fin del programa NWSETUP.PRG
```

Figura 27-4.

Un programa para personalizar las opciones de ámbito global del sistema *(continuación)*.

que NW.MEM no ha sido localizada en el disco, el programa llama al procedimiento DEFAULTS, que establece los valores por omisión para la mayoría de las variables de ámbito global.

Los restantes pasos son llevados a cabo sin tener en cuenta cómo ha sido llamado el programa de configuración. El procedimiento COLORS comprueba el valor de la función ISCOLOR para determinar si dispone de un adaptador de visualización de color/gráficos (CGA, EGA o VGA) y establece los colores de visualización o los atributos monocromáticos. El procedimiento crea ocho variables de memoria, siete de ellas para controlar los colores de los objetos de pantalla que puede personalizar con las órdenes SET COLOR OF y una variable adicional llamada MSWARNING que almacena el color utilizado para los mensajes de aviso drásticos (en este ejemplo, las letras en rojo sobre fondo negro en un monitor de color, o letras brillantes sobre un fondo oscuro en pantallas monocromáticas). Con estas variables se configuran los colores utilizados para todos los objetos explícitamente, incluso si está satisfecho con los valores por omisión de dBASE IV, ya que no hay forma de saber si los usuarios han cambiado los colores ejecutando órdenes SET en el punto indicativo, con los menús o editando el archivo CONFIG.DB, antes de arrancar la aplicación. Más tarde en el programa de configuración, el usuario tiene la oportunidad de anular los colores especificados en el procedimiento COLORS para el texto y los datos introducidos.

Los programas de entrada de datos del Capítulo 24 utilizaban una variable de memoria llamada MPRESSKEY para evitar la repetición del texto de un mensaje ("-Pulse cualquier tecla para continuar"). El procedimiento STDMSGS crea algunas variables adicionales en las que puede almacenar las frases utilizadas en muchos de los mensajes visualizados por el sistema National Widgets. Además, puede añadir variables para almacenar los mensajes utilizados en algunas de sus líneas de menús y menús de aparición súbita. Por ejemplo, podría crear una variable de memoria MSCHOOSE, para el mensaje de la línea de mensaje estándar que advierte al usuario cómo debe elegir una opción de una línea de menús o de una ventana.

```
mschoose = "Para seleccionar una opción, destáquela y pulse <ENTER> " +;
   "o pulse la letra inicial"
```

Luego nombraría esta variable en una orden DEFINE MENU o DEFINE POPUP del siguiente modo:

```
DEFINE POPUP lblmenu FROM 7,26 TO 15,53 MESSAGE mschoose
```

Hasta ahora, el programa de configuración no ha tenido interacción con el usuario. Si los pasos preliminares consumen mucho tiempo de ejecución en su computadora, podría visualizar un mensaje de estado que describa lo que cada procedimiento está haciendo durante la ejecución. A continuación, el programa permite que el usuario personalice las opciones de ámbito global independientes de la aplicación empezando con el estado de los parámetros BELL y CONFIRM. Estas selecciones se almacenan en las variables MSBELL y MSCONFIRM, las cuales tienen el valor "ON" o "OFF" para acomodarse a la sintaxis de las órdenes SET correspondientes.

Puesto que es un error suponer que el usuario está familiarizado con la sintaxis de las órdenes SET, el programa presenta al operador dos respuestas en términos menos técnicos, "sí" o "no", y las almacena en dos variables temporales, MBELL y MCONFIRM. Estas variables deducen su valor inicial del contenido presente de MSBELL y MSCONFIRM; cuando el programa de configuración se ejecuta por primera vez, estos son los valores establecidos en el procedimiento DFAULTS. Nótese que la función IIF que asigna a MBELL su valor inicial da como resultado .T. si el parámetro BELL está en OFF, mientras que MCONFIRM está .T. si MSCONFIRM está en ON. Estas asignaciones le permiten expresar las respuestas presentadas al operador de modo que al responder "sí" a una de las cuestiones el parámetro (BELL) se pone en OFF, mientras que al responder "sí" a la otra el parámetro CONFIRM se pone en ON. Después de la orden READ que capta las entradas del usuario, el programa traduce los valores de MBELL y MCONFIRM en "ON" o "OFF" para suministrar nuevos valores a las variables de opciones de ámbito global MSBELL y MSCONFIRM.

A continuación, el programa permite que el usuario seleccione el formato de visualización de las fechas a partir de la lista de selección DATEFORM definida en el procedimiento MENUDEF (si su aplicación incluye órdenes que dependen de un formato de fecha particular, no puede permitir que los usuarios personalicen este parámetro). Antes de activar la ventana DATEFORM, el programa establece las opciones de ámbito global MSDATE en "AMERICAN" e inicializa la variable de memoria MDATE, que almacena una cadena de caracteres que representa el formato de visualización de fecha con el valor "MM/DD/YY". Si el usuario pulsa ESC para salir de la lista de selección sin hacer ninguna, estos valores por omisión permanecen inactivos, y la aplicación visualiza las fechas en el formato familiar *MM/DD/YY*.

El procedimiento DATEFORM, que procesa la selección de la lista de selección, almacena la cadena de caracteres que representa el formato elegido (el valor de la función PROMPT) en la variable de memoria MDATE y luego utiliza una estructura DO CASE para asignar el valor apropiado a la variable de ámbito global MSDATE según el valor de la función BAR. Por ejemplo, si el usuario selecciona "MM-DD-YY" de la lista de selección, el procedimiento DATEFORM asigna el valor "USA" a MSDATE. Cuando el procedimiento DATEFORM desactiva la ventana DATEFORM, el programa de configuración visualiza el valor de MDATE para recordar al usuario el formato que ha seleccionado. La lista de selección de formatos de fecha incluye solamente seis selecciones, mientras que el manual de referencia de dBASE IV lista 11 formatos de visualización de fecha. De los 11 formatos, solamente 6 son distintos, los otros están duplicados.

La rutina más compleja del programa de configuración permite que los usuarios seleccionen los colores de fondo y primer plano utilizados en toda la aplicación para visualizar el texto y los datos introducidos. Esta secuencia de pasos se ejecuta solamente si dispone de un adaptador de visualización color/gráficos (si la función ISCOLOR da por resultado .T.); pero no es difícil añadir órdenes análogas para seleccionar los atributos de visualización monocromáticos también. Del mismo modo que el menú **Pantalla (Display)** de dBASE IV, el programa de configuración permite que el usuario seleccione los colores de la lista de selección, pero no puede imitar estos menús exactamente y visualizar cada línea en el color descrito, ya que no puede es-

pecificar los colores individualmente para las líneas de una lista de aparición súbita (pop-up).

En su lugar, el programa de configuración visualiza muestras de los colores de fondo y primer plano disponibles en las tres filas de la parte inferior de la ventana y permite que el usuario seleccione cada uno de los colores a partir de una lista de aparición súbita estándar. El procedimiento COLORBAR, que representa las líneas de colores de muestra, visualiza el nombre de cada uno de los colores del primer plano en ese color frente a un fondo negro (excepto "Negro", que se muestra en un fondo blanco). El nombre de los colores de fondo se visualiza en letras negras en una línea del color de fondo especificado (excepto el "Negro" que se visualiza frente a un fondo blanco).

Las listas de selección que espécifican el nombre de los colores son visualizadas por las ventanas FOREGRND y BACKGRND, y las selecciones de la lista de selección se procesan mediante dos procedimientos que tienen el mismo nombre, que almacenan las selecciones del usuario en las variables de memoria MFOREGRND y MBACKGRND. La apariencia de la pantalla con la lista de selección de los colores de primer fondo visualizada se muestra en la Figura 27-5. Cada lista de selección se visualiza dos veces: una para los colores de texto y otra para los colores del campo. Después de activar la lista de selección, el programa inicializa MFOREGRND y MBACKGRND con las abreviaturas de los colores por omisión de la aplicación (blanco sobre azul para el texto y blanco sobre cian para los datos) de modo que si el usuario pulsa ESC para salir de cualquier lista de selección sin hacer ninguna, las subsecuentes órdenes que asignan nuevos valores a MSNORMAL y MSFIELDS mantendrán los parámetros de color anteriores.

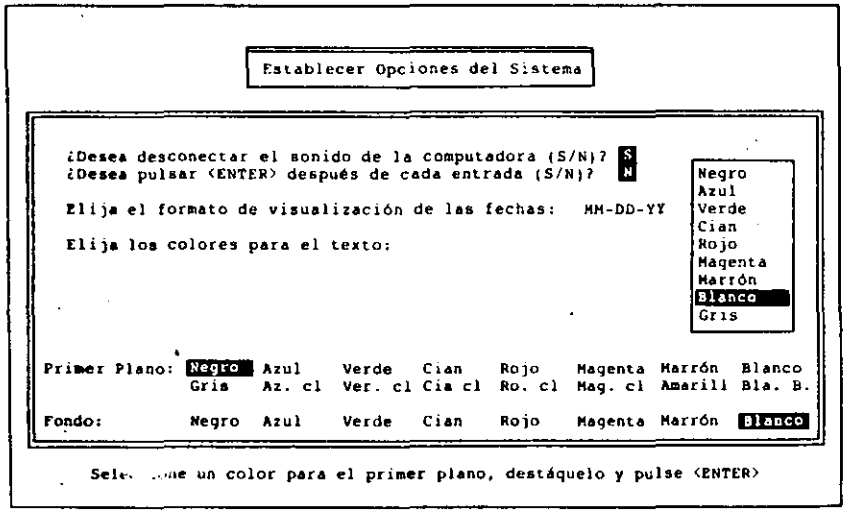

Figura 27-5.

Permite que el usuario seleccione los colores a partir de una lista de selección.

Cada código de color se construye combinando los códigos de fondo y primer plano con los separadores de barra inclinada (/) requeridos entre ellos. Después de crear cada variable, el programa visualiza el texto: "Color del texto" o "Color de los datos" en los colores elegidos para que el usuario compruebe el efecto de la elección. Finalmente, una serie de órdenes SET COLOR activan inmediatamente todas las selecciones de color (aquellas establecidas por el procedimiento DEFAULTS, así como las especificadas por el usuario).

El método utilizado para deducir el código de los colores a partir de las selecciones de la lista, ilustran una complicada pero potente y concisa técnica para enfrentar la selección de una lista a los valores almacenados en una variable de memoria. En este caso, la variable de memoria MCOLORS, inicializada al principio del programa de configuración, almacena una lista con el nombre de los colores (los visualizados en la lista de selección FOREGRND y BACKGRND) y los códigos de color. El nombre del color está encerrado por un par de barras inclinadas, rellenándose con espacios en blanco al final para que todos ellos tengan la misma longitud, 7 caracteres, y cada nombre de color va seguido por el correspondiente código de color de dBASE IV, cumplimentado si es necesario a 3 caracteres con espacios en blanco.

Puede interpretar el contenido de la variable MCOLORS como una tabla de traducción en la que cada código de color se almacena en una posición fija relacionada con el nombre del color correspondiente; cada código empieza 9 caracteres a la derecha de la barra inclinada que precede al nombre del color. Puede utilizar la función SUBSTR para quitar los tres caracteres que representan el código de color de MCOLORS; la posición inicial de la subcadena se localiza con la función AT, que busca el principio del nombre de color correspondiente. Por ejemplo, la siguiente expresión extrae el código que representa al cian de MCOLORS:

```
SUBSTR(mcolors, AT("/Cyan /", mcolors)+9,3)
```

Puede utilizar la misma técnica para quitar la descripción correspondiente a un código corto de una variable análoga a MCOLORS en lugar de almacenar los códigos y descripciones en un archivo tal como el archivo Códigos. El uso de una variable de memoria es apropiado cuando existen relativamente pocos códigos, no cambian con frecuencia, y los usuarios no ncesitan actualizarlos. Las barras inclinadas de MCOLORS hacen que la variable sea más legible y evita la ambigüedad en el nombre de los colores, incluso si la misma combinación de caracteres ocurre en cualquier parte de la variable. En este caso, las barras inclinadas no son necesarias, pero si aplica la misma técnica para extraer la descripción puede necesitarlas.

Los procedimientos FOREGRND y BACKGRND utilizan este método para buscar las selecciones hechas por el usuario de una lista, determinada evaluando la función PROMPT, en MCOLORS y extraer los códigos de color correspondientes. Merecen especial atención unos cuantos detalles adicionales: la función PROMPT siempre evalúa una cadena de caracteres sin espacios en blanco a la derecha, de modo que puede utilizar esta cadena como parte de un mensaje sin recortarla. Para escribir una expresión explícita que dé por resultado la posición inicial del código de color

que sigue al nombre del color, hay que tener en cuenta que el nombre de los colores de MCOLORS son todos iguales en longitud, 7 caracteres. Para extender la cadena que resulta de evaluar la función PROMPT cada procedimiento añade 7 espacios a esa cadena y luego almacena los 7 caracteres de la izquierda del resultado en la variable de memoria MCOLORNAME. Para encontrar la posición inicial del código de color, los procedimientos utilizan la función AT para buscar la expresión "/" + MCOLORNAME + "/" en la variable MCOLORS y suman 9 al resultado. La longitud de la subcadena (el código de color) siempre es de 3 caracteres, y la función TRIM elimina los espacios en blanco de la derecha del código.

El resto del programa de configuración está claro. El programa permite que el usuario seleccione la impresora por omisión de una lista de selección e introduzca la unidad de disco en la que ejecuta la aplicación. Las opciones específicas de la aplicación se colocan en una segunda pantalla, que indica al usuario que introduzca el nombre de la compañía, tres líneas para la dirección, el estado por omisión y el código de área de los neuvos clientes, los impuestos sobre las ventas por omisión, y el siguiente número de factura disponible. El programa valida el estado y el código del área consultando las entradas del usuario en el archivo de Códigos (exactamente igual que el programa de actualización descrito en el Capítulo 24 que validaba entradas de campo comparables en el archivo Clientes). Finalmente, el programa de configuración almacena las nuevas opciones en el archivo de variables de memoria NW.MEM y regresa al programa que llama (el programa de arranque principal de la aplicación o el programa del menú de utilidades).

Uso de las variables de opciones de ámbito global del sistema

Alguna de las variables de opciones de ámbito global del sistema son utilizadas en la versión modificada del programa de arranque de la aplicación para establecer el entorno de trabajo. Por ejemplo, las órdenes SET BELL, SET CONFIRM y SET DATE utilizan las variables MSBELL, MSCONFIRM y MSDATE en lugar de establecer los parámetros del siguiente modo:

```
SET BELL &msbell
SET CONFIRM &msconfirm
SET DATE &msdate
```

Otras variables, incluyendo la del estado por omisión, código de área, impuestos de las ventas, y el siguiente número de factura disponible, que se han usado en el programa de consulta y actualización del sistema y en los programas de informes descritos en el Capítulo 23, imprimen el nombre de la compañía almacenado en MSCOMPANY en el título del informe. Los programas de utilidades descritos más adelante en este capítulo, que copian las bases de datos del sistema en discos flexibles y restauran las bases de datos desde copias de seguridad, utilizan MSDRIVE para especificar la unidad de disco que contiene la aplicación.

En otros casos, el uso de variables requiere modificaciones más exhaustivas de los programas existentes. En particular, los colores de visualización almacenados en

las variables de ámbito global, que se utilizan en las órdenes SET COLOR OF en NW2.PRG, para especificar los colores de visualización generales de la aplicación, también pueden utilizarse en otros programas del sistema. Por ejemplo, cuando llama al procedimiento CENTER, MESSAGE, o MESSAGE2, podría incluir una de las variables para especificar los colores de visualización para el texto del mensaje (la mayoría de los ejemplos presentados hasta ahora pasaban una cadena más al parámetro de color específico, tal como "W + /N", que funciona en cualquier monitor.

Puede utilizar variables de código de color para anular los colores por omisión asignados por dBASE IV a un objeto de la pantalla para que coincida con los colores utilizados por otro. Por ejemplo, podría pasar los códigos de color almacenados en MSBOX al procedimiento SCRNHEAD de modo que dBASE IV represente el contorno que rodea al encabezamiento de la pantalla con los mismos colores utilizados para el marco del contorno de una ventana o menú que aparece en la pantalla al mismo tiempo (por omisión, dBASE IV utiliza los colores especificados en una orden SET COLOR OF NORMAL para representar los recuadros con órdenes @, mientras que utiliza los colores establecidos con SET COLOR OF BOX para los marcos del menú y de la ventana). También puede que desee especificar los colores de visualización utilizados para el texto de encabezamiento, tal como se hace en la siguiente versión del procedimiento SCRNHEAD.

```
PROCEDURE Scrnhead
PARAMETERS mframe, mheading, mtext

mtext = "" + TRIM(mtext) + ""
mtextstart = (80-LEN(TRIM(mtext)))/2
@ 1, mtextstart-1 TO 3, 81-mtextstart 205,196,179,213,184,192,217;
   COLOR &mframe
@ 2, mtextstart SAY mtext COLOR &mheading.

RETURN
```

El Capítulo 16 señaló que dBASE IV utiliza los colores especificados con SET COLOR OF HIGHLIGHT para los datos captados en una ventana con órdenes @ ... GET en lugar de los colores establecidos con SET COLOR OF FIELDS. Podría modificar los programas de consulta y actualización descritos en el Capítulo 24 para visualizar campos en los colores estándar definidos por la orden SET COLOR OF FIELDS en el programa de arranque principal de la aplicación añadiendo una cláusula COLOR a cada orden @ ... GET, como en el siguiente ejemplo:

```
@ 4,15 GET Company COLOR ,&msfields.
```

La cláusula COLOR puede incluir dos pares de códigos de color, el primero especifica los colores utilizados para el texto visualizado, y el segundo, los colores utilizados para los datos introducidos. Si omite el primer código, debe incluir la coma que separa los dos para evitar que dBASE IV interprete sus códigos de color como colores del texto. En una orden @ ... SAY ... GET puede listar ambos pares de códigos de colores del siguiente modo:

```
@ 4, 2 SAY "Company" GET Company COLOR &msnormal., &msfields.
```

De forma similar, las variables de memoria que almacenan los mensajes estándar definidos en el procedimiento STDMSGS en el programa de configuración podrían utilizarse en todas las partes de la aplicación. Podría construir muchos de los mensajes de error y mensajes de información visualizados en el programa de consulta y actualizacuión del archivo de Clientes combinando esas variables con cadenas de caracteres constantes que expresen segmentos únicos de los mensajes. Por ejemplo, podría volver a escribir la estructura IF que visualiza un mensaje de error si el usuario selecciona la opción **Anterior** cuando el puntero de registro esta posicionado todavía en el primer cliente, tal como sigue:

```
IF BOF()
   DO Message3 WITH msfirst + "customer"
   GOTO TOP
ENDIF
```

Podría utilizar la variable MSONFILE en la orden @... GET que capta y valida los códigos de cuenta asignados a un nuevo cliente del siguiente modo:

```
@ 1,2 SAY "Introduzca un código de cuenta único para el nuevo cliente";
      SET maccount PICTURE "@A!";
      VALID maccount = "" OR. Isunique(maccount);
      ERROR TRIM(maccount) + msonfile
```

Incluso si piensa que no merece la pena editar los programas de una aplicación existente para personalizar lo colores utilizados para cada @... GET, puede incorporar fácilmente esta técnica en las nuevas aplicaciones que diseñe.

ACCESO A LOS PROGRAMAS Y ORDENES DEL DOS

Cuando trabaja desde el punto indicativo, es conveniente que sea capaz de realizar tareas de gestión de archivos tales como visualización de listas de archivos, copias, cambios de nombre, y comprobación del espacio de disco disponible sin salir de dBASE IV. Aunque dBASE IV proporciona aproximaciones equivalentes para las órdenes DIR, RENAME, ERASE, COPY, y TYPE del MS-DOS, ellas difieren en la sinfaxis y las posibilidades y en la mayoría de los casos, están más limitadas. En particular, las órdenes RENAME, ERASE, y COPY de dBASE IV no permiten que utilice los caracteres ? y * como símbolos "comodín" reconocidos por MS-DOS, de modo que no puede utilizar una sola orden para operar sobre un grupo de archivos con nombres similares.

La introducción a la programación de dBASE IV del Capítulo 9 estableció que podía utilizar un editor de texto externo sin salir de dBASE IV, con tal de que tenga suficiente memoria RAM en su computadora. De hecho puede utilizar la orden RUN para ejecutar cualquier orden intrínseca del DOS, archivo batch (archivo BAT), o programa (archivo .COM o .EXE) desde el punto indicativo o desde dentro de un programa de dBASE IV. A diferencia de otros programas de tratamiento de textos, programas tales como las utilidades CHKDSK, BACKUP, RESTORE y FORMAT del DOS son bastante pequeñas para ser ejecutadas desde dentro de dBASE IV en cualquier sistema de 640K. Necesita bastante memoria para cargar el COMMAND.COM (entre 18K y 40K, dependiendo de su versión del DOS) y los programas externos, y bastante memoria RAM adicional para el programa (órdenes incorporadas tales como DIR, ERASE, RENAME, y COPY no requieren memoria adicional). La sintaxis de la orden RUN es:

RUN *nombre del programa u orden del DOS*

o

!nombre del programa u orden del DOS

Recuerde que tanto el COMMAND.COM como el programa que desea ejecutar deben estar disponibles para dBASE IV cuando ejecuta la orden RUN. El DOS busca el COMMAND.COM en el directorio raiz del disco. Si arranca su sistema de disco fijo desde un disco flexible puede mantener el disco de arranque en la unidad o puede copiar el COMMAND.COM en el disco fijo y utilizar la orden SET COMSPEC en su archivo AUTOEXEC.BAT (esta es una orden del entorno del DOS, *no* una orden SET de dBASE IV) para especificar su nueva posición. Por ejemplo, si el COMMAND.COM reside en un subdirectorio llamado DOS en la unidad C, utilizaría la siguiente orden en el archivo AUTOEXEC.BAT:

```
SET COMSPEC=C:\DOS\COMMAND.COM
```

dBASE IV también debe ser capaz de encontrar el programa externo o el archivo batch que desea ejecutar y, en la mayoría de los casos, este archivo no está localizado en el mismo subdirectorio que su aplicación. Una forma de acceder al archivo desde otro subdirectorio es incluir el nombre del camino completo en la orden RUN, como en el siguiente ejemplo:

```
RUN C:\DOS\CHKDSK
```

Un método mejor, que es menos dependiente de la configuración hardware y de la estructura del subdirectorio, implica el uso de la orden PATH del DOS antes de cargar dBASE IV para establecer un camino de búsqueda por omisión para los archivos que no se encuentran en el subdirectorio actual. Si la orden RUN no incluye el nombre del camino, operará sobre el disco fijo, con tal de que esté activa la PATH del DOS apropiada. Si desea utilizar el mismo camino de búsqueda, puede incluir la orden PATH antes de cargar dBASE IV o incluirla en el archivo batch utilizado para arrancar dBASE IV. El camino de búsqueda del DOS puede incluir otros subdirectorios distintos del que contiene los programas de utilidad a los que necesita acceder desde dBASE IV. Por ejemplo, para acceder a los programas de utilidad almacena-

dos en los directorios DOS y UTILITY, y poder ejecutar dBASE IV o WordPerfect desde cualquier subdirectorio, utilizaría la siguiente orden PATH:

```
PATH C:\DOS;C:\UTILITY;C:\DBASE4;C:\WP
```

Para usuarios más avanzados que trabajan a nivel de mandato, la posibilidad de ejecutar órdenes de DOS desde el punto indicativo es mucho más conveniente. En un programa de dBASE IV, la orden RUN tiene una gran potencia: le permite dar acceso a las órdenes del DOS en un entorno dirigido por menú con un interfaz con el usuario que es consistente con el resto de su aplicación. El número de órdenes DOS que deberían incorporarse dentro del sistema depende en gran medida de la experiencia y las necesidades de los usuarios. Al menos debería proporcionar opciones para hacer copias de seguridad de las bases de datos y restaurar los archivos a partir de dichas copias en el caso de que un sistema falle. Para facilitar este proceso, puede permitir que los usuarios formateen discos o verifiquen el espacio de disco disponible sin salir al sistema operativo. Colocar la opción para hacer copias de seguridad en uno de los menús del sistema disminuye la probabilidad de que esta importante tarea sea olvidada, en parte debido a que la ejecución desde un menú es bastante más fácil para quien lo hace, y en parte debido a que al verla en el menú sirve como un recordatorio diario.

A menos que sólo necesite unas pocas órdenes del DOS en una aplicación (digamos, la opción restaurar y hacer copias de seguridad) el mejor sitio para colocarlas es en un menú de utilidades, tal como el que se muestra en la Figura 27-2. Dos de las opciones son simples: la opción **Comprobar el espacio del disco** llama a la orden CHKDSK del DOS y la opción **Formatear discos** llama a la utilidad FORMAT. En ambos casos, el programa borra la pantalla y ejecuta la orden apropiada del DOS con una orden RUN. Cuando utilice la orden RUN en un programa, dBASE IV se detiene automáticamente y visualiza el mensaje "Pulse cualquier tecla para continuar..." después de ejecutar la orden, de modo que no necesita añadir una orden WAIT para que el usuario tenga tiempo de leer la pantalla.

Un programa para ejecutar las órdenes del DOS especificadas por el usuario

Por conveniencia de los usuarios más experimentados con su aplicación, también puede proporcionar una opción para introducir y ejecutar cualquier programa u orden del DOS desde dentro del sistema de menús de la aplicación. Puede hacer esto almacenando la orden del usuario en una variable de memoria y ejecutarla extendiendo la variable como una macro en una orden RUN. La Figura 27-6 lista un programa llamado NWRUNDOS.PRG que ilustra esta técnica. Puesto que el usuario puede que desee ejecutar más de una orden del DOS antes de volver al menú, el programa establece un bucle "indefinido" con DO WHILE .T.

El programa define una pequeña ventana llamada RUNDOS para visualizar el mensaje y captar la orden. A diferencia de todos los ejemplos presentados hasta ahora, el programa NWRUNDOS activa y desactiva la ventana cada paso a través del bucle


```
* NWRUNDOS.PRG
* PROGRAMA PARA EJECUTAR ORDENES DEL DOS ESPECIFICADAS POR EL USUARIO
* ULTIMA REVISION: 07/31/89   M.LISKIN

DEFINE WINDOW rundos FROM 8,10 TO 14,70 DOUBLE

DO WHILE .T.
   CLEAR
   ACTIVATE WINDOW rundos
   mcommand = SPACE(100)              && Orden del DOS
   DO Center WITH 1, 60, "",;
      "Introduzca una orden del DOS, o pulse <ENTER> para salir"
   @ 3, 5 GET mcommand FUNCTION "S50"
   READ
   DEACTIVATE WINDOW rundos
   IF mcommand = ""                   && Si el usuario deja en blanco la orden
      EXIT                                    && sale del programa
   ELSE
      RUN &mcommand.
   ENDIF
ENDDO

RELEASE WINDOW rundos
RETURN

* Fin del programa NWRUNDOS.PRG
```

Figura 27-6.

Un programa que ejecuta una orden del DOS especificada por el usuario.

DO WHILE. Puede pensar en un principio que esto es necesario porque sería inapropiado limitar la ejecución de la orden DOS a la ventana pequeña, pero esto es un punto discutible. Cuando dBASE IV ejecuta una orden RUN carga una nueva copia del COMMAND.COM en la memoria y suspende temporalmente la operación de la copia del DOS bajo la que se ejecuta dBASE IV. Esta nueva copia del sistema operativo no puede tener en cuenta ningún aspecto del entorno establecido por dBASE IV bajo el control del procesador de órdenes original, y dirige su salida a la pantalla completa. Cuando la orden del DOS termina, la segunda copia del COMMAND.COM es descargada y dBASE IV toma el control de nuevo. En este punto, si la primera ventana estuviera todavía activa, volvería a aparecer en la pantalla con dos efectos no deseables: cubriría la salida del programa del DOS, posiblemente oscureciendo el texto que el usuario no ha leído todavía, y no le permitiría borrar esta salida de la pantalla antes de captar la siguiente orden DOS.

El programa NWRUNDOS borra la pantalla, activa la ventana RUNDOS al principio del bucle DO WHILE, y la desactiva antes de ejecutar cada orden DOS. La

variable de memoria que almacena la orden del usuario está inicializada con 100 espacios en blanco y es recogida con una orden @ ... GET que utiliza la función de imagen S para desplazar horizontalmente las entradas que son más largas que la región de entrada de datos de 50 caracteres. Si el usuario deja MCOMMAND en blanco, el programa sale del bucle DO WHILE y regresa al programa que llama. En cualquier otro caso, ejecuta la orden DOS del usuario con una orden RUM.

El procedimiento.RUNDOS no intenta comprobar o validar la línea de orden del DOS del usuario, ya que no es posible determinar si es una orden del DOS apropiada o válida. Dar al usuario esta libertad no entraña un mayor riesgo de dañar las bases de datos, ya que el usuario accede a las mismas órdenes desde el punto indicativo o desde el DOS fuera de dBASE IV. Aunque es posible de hecho introducir un orden destructiva tal como ERASE *.DBF, es improbable que un error sintáctico de mecanografía dañe la integridad de las bases de datos. La introducción de una línea de orden errónea dará como resultado, en la mayoría de los casos, el mensaje de error estándar del MS-DOS "Bad command o filename". En el peor de los casos, si la ejecución de un programa incompleto hace que el sistema realice una "pasada imprevista" es más seguro volver a arrancar o desconectar la computadora, ya que no hay bases de datos abiertas.

Programas para hacer copias de seguridad de las bases de datos

El Capítulo 26 describió cómo utilizar el catálogo de archivos de dBASE IV para escribir programas de utilidad que no dependieran de los nombres de los archivos utilizados en la aplicación. Puede aplicar la misma técnica para escribir programas de copias de seguridad de uso general, que hagan uso del catálogo del sistema para identificar las bases de datos del sistema y estimar su tamaño en conjunto. La Figura 27-7 lista un programa denominado NWBACKUP.PRG que realiza copias de seguridad de los archivos de variables de memoria y bases de datos (archivos .DBF y .DBT) de la aplicación, los archivos de índices pueden ser reconstruidos, si es necesario, utilizando uno de los programas de reindexación descritos en el Capítulo 26.

Para asegurar que el usuario no arranca el procedimiento para hacer copias de seguridad sin disponer de suficientes discos flexibles formateados para hacerlo, NWBACKUP.PRG estima el tamaño de todos los archivos de bases de datos del sistema e informa al operador del número de discos necesarios (si no utiliza campos memos, el cálculo será estimado de forma más exacta). Como se describió en el Capítulo 2, que resumía cómo estimar el espacio total de disco requerido para acomodar la aplicación, el tamaño de una base de datos de dBASE IV es aproximadamente igual al número de registros presentes en el archivo multiplicado por la longitud del registro. Estos números pueden obtenerse mediante las funciones RECCOUNT(), que da por resultado el número de registros actuales, y RECSIZE() que da la longitud del registro. Para estimar el tamaño de un archivo .DBT asociado con cualquier base de datos que contenga campos memo, el programa NWBACKUP.PRG supone que el tamaño de cada campo memo es de 1024 bytes (dos bloques de 512 bytes). Puede que necesite ajustar esta cifra en alguna de sus aplicaciones.


```
* NWBACKUP.PRG
* PROGRAMA PARA HACER COPIAS DE SEGURIDAD DE LAS BASES DE DATOS
* ULTIMA REVISION: 08/03/89   M.LISKIN

DEFINE WINDOW utility FROM 4, 10 TO 19,70
ACTIVATE WINDOW utility

DO Center WITH 5, 60, "",;
   "Calculando el tamaño de los archivos --Por favor no interrumpa"

mfilesize = 3000                && Tamaño estimado del archivo MEM

SELECT Filelist                 && Area de trabajo del catálogo
SET FILTER TO UPPER(Type) = "DBF" && Selecciona las entradas de bases de datos
SCAN
   SELECT 2
   USE (Filelist -> Path)               && Abre una base de datos
   @ 7, 0
   DO Center WITH 7, 60, "", "Trabajando con " + DBF()
   * Cuenta los campos memo y calcula el tamaño de la cabecera
   mcount = 1                           && Número de campos
   mmemos = 0                           && Número de campos memo
   mheader = 34                         && Tamaño de la cabecera del archivo
   DO WHILE LEN(FIELD(mcount)) > 0      && Para cada campo del archivo,
      IF TYPE (FIELD(mcount)) = "M"     && Si el campo es un campo memo
         mmemos = mmemos + 1            && Incrementa el contador memo
      ENDIF
      mheader = mheader + 32            && Actualiza el tamaño de la cabecera
      mcount = mcount + 1               && Incrementa el contador de campo
   ENNDO
   * Tamaño estimado de la base de datos: cabecera + datos + texto memo estimado
   mfilesize = mfilesize + mheader + RECCOUNT() * RECSIZE() +;
      RECCOUNT() * mmemos * 1024
   SELECT Filelist
ENDSCAN

* Calcula el número de discos de 360K y 1,2M disks necesarios para la copia de
seguridad
mdisk360 = INT(mfilesize / 360000) + 1 && Número de discos de 360K necesarios
mdisk1200 = INT(mfilesize / 120000) + 1 && Número de discos de 1.2M necesarios

* Establece los márgenes para el mensaje de aviso
_lmargin = 2
_rmargin = 58
_wrap = .T.
```

Figura 27-7.

Programa para hacer copias de seguridad de las bases de datos.

```
CLEAR
?  "Necesitará"
?? LTRIM(STR(mdisk360,3))
?? " discos de 360K o"
?? LTRIM(STR(mdisk 1200,3))
?? "de 1.2M para hacer la copia de seguridad de las bases de datos."
?
?  "Si no dispone de suficientes discos formateados, responda "
?? '"N" (de "No") a la pregunta "¿Está preparado para hacer las copias de se-
    guridad'
?? 'de las bases de datos?'"
?
?  "Puede formatear los discos seleccionando la opción Formatear del menú "
?? "Utilidadesopuedesalirdeestaaplicaciónyejecutarelprogramaformat "
?? "desde el indicador del DOS."
?
mconfirm = .F.
@ 12, 5 SAY "¿Está preparado para hacer la copia de seguridad de las bases de
datos? (Y/N)"
        GET mconfirm PICTURE "Y"
READ
DEACTIVATE WINDOW utility

IF mconfirm
   RUN BACKUP &msdrive.*.MEM A:
   RUN BACKUP &msdrive.*.DBF A: /A
   RUN BACKUP &msdrive.*.DBT A: /A
   msbackup = date()                          && Almacena la fecha de la última
                                                 copia de seguridad
   SAVE ALL LIKE ms* TO &msprefix.
   WAIT "Copia de seguridad terminada -- Pulse cualquier tecla para continuar"
ENDIF

_lmargin = 0
_rmargin = 80
RELEASE WINDOW utility

RETURN

* Fin del programa NWBACKUP.PRG
```

Figura 27-7.

Programa para hacer copias de seguridad de las bases de datos *(continuación)*.

Realmente una base de datos es algo más extensa que el número obtenido de multiplicar RECCOUNT() por RECSIZE(). dBASE IV almacena una descripción de la estructura del archivo al principio del archivo .DBF. Esta cabecera consta de 32 bytes de información que describen cada uno de los campos de la base de datos, así como una parte constante de 34 bytes de longitud, que contiene la última fecha de actualización, el tamaño del registro, el número de registros del archivo, y otras informaciones utilizadas internamente por dBASE IV. Puede calcular el tamaño exacto de la cabecera multiplicando el número de campos por 32 y sumando 34 al resultado.

Aunque no existe una función que proporcione directamente el número de campos en la estructura, puede utilizar la función FIELD, que acepta un número como entrada y devuelve como salida el nombre del campo que ocupa la posición especificada en la estructura, o una cadena nula si no existe tal campo. Por ejemplo, en el archivo Clientes de National Widgets, FIELD(1) da por resultado la cadena de caracteres "ACCOUNT" y FIELD(50) da por resultado una cadena nula. El programa NWBACKUP.PRG debe comprobar cada campo a la vez que cuenta los campos memo, y es bastante sencillo calcular el tamaño de la cabecera en el mismo instante. Si no utiliza campos memo, no necesita examinar cada campo, y para acelerar los cálculos podría suponer un tamaño fijo, digamos 3000 bytes, para la cabecera del archivo. Para contar el tamaño del archivo de variables de memoria, NW.MEM, se asigna a la variable que acumula el tamaño total del archivo, MFILESIZE, un valor inicial de 3000 bytes, una estimación que es suficiente para la mayoría de las aplicaciones.

Al igual que los programas de reindexación descritos en el Capítulo 26, el programa de copia de seguridad supone que el archivo de catálogo del sistema está abierto y que su alias es FILELIST. Abriendo el catálogo con una orden USE en el programa del menú de utilidades puede asegurar que el procedimiento de copias de seguridad no contenga nombres de archivos específicos de la aplicación y pueda ser utilizado en cualquier sistema. El programa selecciona el área de trabajo del catálogo, establece un filtro basado en una condición que selecciona solamente las entradas del catálogo que representan archivos de bases de datos, y utiliza un bucle SCAN para examinar las entradas de la base de datos del catálogo. Dentro del bucle SCAN, el procedimiento selecciona una segunda área de trabajo, abre el archivo descrito por el registro actual en el archivo de catálogo usando paréntesis para la sustitución del nombre del archivo, y visualiza un mensaje de estado que incluye el nombre del archivo que está actualmente en proceso, expresado con la función DBF.

A continuación, el procedimiento utiliza un bucle DO WHILE para comprobar los campos de uno en uno, calcula el tamaño de la cabecera del archivo, y cuenta el número de campos memos en la estructura. La variable de memoria MCOUNT cuenta el número total de campos, MMEMOS almacena el número de campos memo, y MHEADER acumula el tamaño de la cabecera. La condición de la sentencia DO WHILE da por resultado .T. hasta que MCOUNT es mayor que el número de campos de la estructura, y la función FIELD da por resultado una cadena nula. Para cada campo, el programa añade 32 a MHEADER y comprueba el valor de la función TYPE para determinar si el campo es un campo memo. La función TYPE da por resultado una letra que representa el tipo de datos de la expresión suministrada como entrada; si el decimoquinto campo de la base de datos es un campo memo, TYPE(FIELD(15)) devuelve el valor "M". Para todos los campos memo que encuentra, el programa incrementa MMEMOS.

Cuando finaliza el bucle DO WHILE, MHEADER contiene el tamaño de la cabecera del archivo y MMEMOS almacena el número de campos memo de la base de datos. El tamaño total del archivo puede ser estimado sumando el tamaño de la ca-

becera, el espacio ocupado por los registros de datos del archivo .DBF (el número de registros multiplicado por el tamaño del registro), y el tamaño aproximado del archivo .DBT (obtenido multiplicando el número de registros, el número de memos, y el tamaño medio de cada memo).

Finalmente, el programa calcula el número de discos de 360K y 1.2M necesarios para hacer las copias de seguridad de todos los archivos y almacena el resultado en las variables de memoria MDISK360 y MDISK1200, respectivamente. El resultado de dividir MFILESIZE por la capacidad de un disco (360.000 ó 1.200.000) es casi siempre una fracción del número de discos, por ejemplo, 4.38 discos. En consecuencia, el programa utiliza la función INT para tomar la parte entera del resultado de la división y luego añade 1 a este número. Si MBYTES / 360000 es igual a 4.38, la función INT da 4, y al añadir 1 se obtiene el total de discos necesarios, 5.

Luego, el programa visualiza el número de discos necesarios para hacer la copia de seguridad con un mensaje que expresa la existencia de otra opción en el menú de utilidades para formatear los discos sin abandonar la aplicación. El mensaje de advertencia se visualiza en una ventana, y el programa de seguridad utiliza las mismas técnicas descritas en el Capítulo 24 para ajustar los márgenes y visualizar los mensajes dentro de los límites de la ventana. En este programa, no es necesario reservar los parámetros de márgenes anteriores antes de volver a ajustar _LMARGIN y _RMARGIN (estas variables se repondrán más tarde en sus parámetros estándar, 0 y 80, anteriormente en efecto).

Después de visualizar el mensaje de advertencia, el programa NWBACK.PRG pide confirmación al usuario para proceder con la copia, y si el usuario contesta afirmativamente, utiliza tres órdenes RUN BACKUP para hacer las copias de seguridad de los archivos (la opción /A de la segunda y tercera orden instruye al DOS para que añada a cada uno de los nuevos grupos de archivos los mismos discos de copia de seguridad en lugar de empezar un nuevo conjunto). Si utiliza una utilidad tal como FastBack en lugar del programa BACKUP del DOS, o si hace una copia de los datos en cintas magnéticas en lugar de en discos flexibles, podría sustituir las órdenes que ejecutan los programas de utilidad apropiados por las órdenes BACKUP de este ejemplo.

El nombre de los archivos se especifica expresando las variables de opciones de ámbito global MSDRIVE y MSPREFIX (que fueron definidas en el programa de configuración descrito anteriormente en este capítulo) con el símbolo de sustitución de macro. En el sistema National Widgets la orden BACKUP es:

```
RUN BACKUP C:NW*.MEM A:
RUN BACKUP C:NW*.DBF A: /A
RUN BACKUP C:NW*.DBT A: /A
```

Este método le evita el tener que escribir el nombre de un archivo específico de la aplicación de forma explícita en órdenes del DOS. Si sabe que no ha sido almacenada ninguna otra base de datos en el mismo subdirectorio de su aplicación, podría hacer copia de seguridad de todos los archivos de variables de memoria y bases de datos con las siguientes órdenes:


```
RUN BACKUP &msdrive.:*.MEM A:
RUN BACKUP &msdrive.:*.DBF A: /A
RUN BACKUP &msdrive.:*.DBT A: /A
```

Cuando termina la copia de seguridad, el programa almacena la fecha actual en la variable de memoria MSBACKUP y almacena las variables de opciones de ámbito global del sistema (incluyendo MSBACKUP) en el disco. La orden SAVE supone que el archivo de variable de memoria que almacena las opciones de ámbito global tiene un nombre que consta del prefijo de la aplicación, y ésta especifica el nombre del archivo expresando la variable MSPREFIX con paréntesis para denotar la sustitución de macro. La versión actual del sistema National Widgets no hace uso de esta fecha. En sus propias aplicaciones, puede visualizar esta fecha en la pantalla de presentación de la aplicación o encima del menú principal o del menú de utilidades. Para los casos en los que la copia de seguridad es crucial, puede comprobar MSBACKUP en el programa de arranque principal de la aplicación y llamar al programa que hace la copia automáticamente si los usuarios arrancan el sistema después de que haya transcurrido demasiado tiempo desde que se hizo la última copia de seguridad.

Podría modificar fácilmente el programa de copia de seguridad para visualizar una lista del nombre de los archivos y permitir que el usuario seleccione los archivos a copiar. Una técnica para hacer esto implica la consulta del usuario cuando el programa examina las entradas del catálogo para calcular el tamaño total de las bases de datos, y marcar los registros del catálogo que describen archivos de las elecciones del usuario para incluirlos en la copia de seguridad. La forma más fácil de marcar las entradas del catálogo es almacenar un símbolo, una marca en la etiqueta TAG, un campo de carácter de longitud 4 proporcionado para su propio uso. La Figura 27-8 lista una nueva versión del programa para hacer copias de seguridad, NWBACK2.PRG que ilustra este método. Podría aplicar la misma técnica para modificar los progra-

```
* NWBACK2.PRG
* PROGRAMA PARA HACER COPIAS DE SEGURIDAD DE BASES DE DATOS SELECCIONADAS
* ULTIMA REVISION: 08/03/89 M. LISKIN

DEFINE WINDOW utility FROM 4,10 TO 19,70
ACTIVATE WINDOW utility

mfilesize = 3000                        && Estima el tamaño del archivo MEM

SELECT Filelist                         && Area de trabajo del catálogo
SET FILTER TO UPPER(Type) = "DBF"       && Selecciona las entradas de la base de
                                           datos
REPLACE ALL Tag WITH " "                && Repone la señal de copia
@ 1,10 SAY "Por favor seleccione los archivos a copiar:"
mline = 3                               && Linea actual de la pantalla
```

Figura 27-8.

Programa para hacer copias de seguridad de bases de datos seleccionadas.

```
SCAN
   IF mline = 13                    && Si la pantalla está completa,
      @ 3,5 CLEAR                   &&    borra el área de visualización
      mline = 3                     &&    repone la línea de la pantalla
   ENDIF
   mbackup = .T.                    && ¿Copia de seguridad de este archivo?
   @ mline, 5 SAY "Copia de seguridad " + TRIM(LEFT(File_name + " (" + Title,
     40));
      + ")? (Y/N)"
   @ mline,55 GET mbackup PICTURE "Y"
   READ
   mline = mline + 1                && Incrementa la línea de la pantalla
   IF mbackup                       && Si el usuario decide hacer la copia,
      REPLACE Tag WITH "B"          &&    marca la entrada del catálogo
      SELECT 2
      USE (filelist->Path)          && Abre una base de datos
      * Cuenta los campos memo y calcula el tamaño de la cabecera del archivo
      mcount = 1                    && Número de campos
      mmemos = 0                    && Número de campos memo
      mheader = 34                  && Tamaño de la cabecera del archivo
      DO WHILE LEN(FIELD(mcount)) > 0 && Para cada campo del archivo
         IF TYPE(FIELD(mcount)) = "M" && Si el campo es un campo memo,
            mmemos = mmemos + 1     &&    incrementa el contador del campo
         ENDIF
         mheader = mheader + 32     && Actualiza el tamaño de la cabecera
         mcount = mcount + 1        && Incrementa el contador del campo
      ENDDO
      * Estima el tamaño de la base de datos: cabecera + datos + texto memo
        estimado
      mfilesize = mfilesize + mheader + RECCOUNT() * RECSIZE() +;
         RECCOUNT() * mmemos * 1024
      SELECT Filelist
   ENDIF
ENDSCAN


* Calcula el número de discos de 360K y 1.2M necesarios para la copia de segu-
ridad
mdisk360 = INT(mfilesize / 360000) + 1       && Número de discos de 360K
mdisk1200 = INT(mfilesize / 1200000) + 1     && Número de discos de 1.2M


* Establece los márgenes para el mensaje de aviso
_lmargin = 2
_rmargin = 50
_wrap = T
```

Figura 27-8.

Programa para hacer copias de seguridad de bases de datos seleccionadas *(continuación)*.


```
CLEAR
?  "Necesitará "
?? LTRIM(STR(mdisk360,3))
?? " discos de 360K o "
?? LTRIM(STR(mdisk1200,3))
?? " de 1.2M para hacer copias de seguridad de las bases de datos."
?
?  "Si no dispone de suficientes discos formateados, responda "
?? '"N" (de "No") a la pregunta "¿Está preparado para hacer las copias de se-
guridad '
?? 'de las bases de datos?'"
?
?  "Puede formatear los discos seleccionando la opción Formatear del menú "
?? "Utilidades o puede salir de esta aplicación y ejecutar el programa FOR-
MAT."
?? "desde el indicador del DOS."
?
mconfirm = .F.
@ 12, 5 SAY "¿Está preparado para hacer la copia de seguridad de las bases de
datos? (Y/N)"
      GET mconfirm PICTURE "Y"
READ
DEACTIVATE WINDOW utility

IF mconfirm
   RUN BACKUP &mprefix.*.MEM A:
   SELECT Filelist
   SCAN FOR Tag = "B"
      mfile = LEFT(TRIM(Path),LEN(TRIM(Path))-1) + "?"
      RUN BACKUP &mfile A: /A
   ENDSCAN
   msbackup = date()                          && Almacena la fecha de la última
                                                 copia de seguridad
   SAVE ALL LIKE MS* TO &msprefix.
   WAIT "Copia de seguridad terminada -- Por favor pulse cualquier tecla para
continuar"
ENDIF

_lmargin = 0
_rmargin = 80
RELEASE WINDOW utility

RETURN

* Fin del programa NWBACK2.PRG
```

Figura 27-8.

Programa para hacer copias de seguridad de bases de datos seleccionadas *(continuación)*.

mas de utilidad descritos en el Capítulo 26; es conveniente que sea capaz de reindexar los archivos de forma selectiva y en el Capítulo 29 se describe un programa para tal fin.

El programa NWBACK2.PRG supone que no utiliza el campo TAG para almacenar ninguna informamción de la base de datos permanentemente y empieza poniendo en blanco este campo con una orden REPLACE en todos los registros del catálogo que representan bases de datos. Para todas las bases de datos que el usuario selecciona para copiar, el programa sustituye el campo TAG con el código "B" (de "backup"). En algunas aplicaciones, puede utilizar el campo TAG para registrar aspectos de las bases de datos que son constantes, pero específicos de la aplicación, tales como una estimación del tamaño medio de un memo. Si este fuera el caso, puede dedicar parte del campo TAG para marcas temporales. Por ejemplo, podría sustituir el último carácter con la señal "B" utilizando la siguiente orden:

```
REPLACE Tag WITH LEFT(Tag,3 + "B")
```

El programa NWBACK2.PRG visualiza el nombre de los archivos uno por uno, utilizando 10 de las 14 líneas de la ventana UTILITY. La variable de memoria MLINE, que está inicializada con el valor 3 y se incrementa después de visualizar cada nombre de archivo de base de datos, sigue la pista de la siguiente línea disponible. Después de pedir que el usuario realice la copia de seguridad de cada archivo, el programa comprueba si MLINE ha alcanzado el valor 13 (la última línea d ela ventana), y si es así, borra las 10 últimas líneas de la ventana (dejando el mensaje "Por favor seleccione los archivos a copiar: " intacto) y empieza de nuevo en la línea 3.

Para cada una de las bases de datos representadas en el catálogo, el programa visualiza una pregunta que incluye el nombre del archivo (el contenido del campo FILE_NAME del catálogo), más, entre paréntesis, tantas descripciones opcionales almacenadas en el campo TITLE como quepan. Para construir esta expresión, el programa combina el campo FILE_NAME, los paréntesis abiertos, y el campo TITLE, y utiliza la función LEFT para extraer los 40 primeros caracteres de la cadena resultante. Luego elimina los espacios en blanco de la izquierda de esta cadena con la función TIRM (en caso de que TITLE sea muy corto) después de añadir los paréntesis de cierre, pedir las marcas, e indicar "(Y/N)" que recuerdan al usuario las dos posibles respuestas a la pregunta.

El programa recoge la respuesta del usuario en una variable de memoria lógica MBACKUP, y si MBACKUP es .T., coloca la marca de la copia, "B", en el campo TAG y añade el tamaño de la base de datos en MFILESIZE, exactamente como en la versión anterior del programa. Después de terminar con los cálculos, que incluyen solamente aquellos campos que el usuario selecciona para copiar, calcula y visualiza el número de discos necesario y pide confirmación para proceder.

El proceso de copia de seguridad también se lleva a cabo en el bucle SCAN que vuelve a leer el archivo de catálogo y selecciona aquellos registros cuyo campo TAG contiene la señal "B". Para cada uno de esos archivos, el campo PATH contiene el nombre del camino completo del archivo .DBF. En caso de que exista también un archivo .DBT, el programa construye el esquema de nombre de archivo del DOS a usar en la orden BACKUP utilizando la función LEFT para eliminar los últimos caracteres del campo PATH y sustituirlos por una "?". La segunda entrada de la función LEFT, que especifica la longitud de la subcadena deseada, se expresa utilizando la función TRIM para eliminar los espacios en blanco del campo PATH, medir la longitud del campo con la función LEN, y sustraer 1 a este número. Para el archivo Clientes de National Widgets, la especificación del archivo final almacenado en la variable de memoria MFILE sería "C:NWCUST.DB?", que describe a los archivos .DBF y .DBT.

Un programa para reponer las bases de datos desde copias de seguridad

La opción **Restaurar de copias de seguridad** del menú Utilidades del sistema National Widgets es llevada a cabo por el programa NWRESTOR.PRG listado en la Figura 27-9. Esta opción permite que cualquier operador recupere bases de datos desde un disco sin tener que comprender la utilidad RESTORE del DOS, pero existe un peligro potencial: RESTORE puede copiar una versión obsoleta del archivo de bases de datos en el disco fijo y escribir encima de los registros actuales. Para reducir la probabilidad de este desastre, el programa da un mensaje de advertencia que explica su propósito y previene al usuario de posibles riesgos. Además, pide confirmación antes de ejecutar la orden RESTORE para asegurarse de que el usuario comprende claramente las consecuencias de esta orden destructiva y permitir deshacer la operación si el usuario hubiera seleccionado la opción por error.

Al igual que en los programas de copias de seguridad, la unidad de disco de la aplicación se especifica expresando la variable de opciones de ámbito global MSDRIVE como una macro, en lugar de incluir una letra de unidad explícita. Esta versión de programa supone que el usuario no hará copias de seguridad de los archivos de índices de la aplicación y visualiza una advertencia que indica que para usar las bases

```
* NWRESTOR.PRG
* PROGRAMA PARA RESTAURAR BASES DE DATOS DESDE COPIAS DE SEGURIDAD
* ULTIMA REVISION: 08/04/89 M. LISKIN

DEFINED WINDOW utility FROM 4,10 TO 19,70
ACTIVATE WINDOW utility

* Repone los márgenes para el mensaje de advertencia
_lmargin = 2
_rmargin = 58
_wrap = .T.
?
```

Figura 27-9.

Programa para restaurar bases de datos a partir de copias de seguridad.

```
DO Center2 WITH 60, "B", "***PRECAUCION***"
?
?  "Este programa restaura bases de datos a partir de copias de seguridad en
discos. "
?? "Los archivos de estas copias escribirán encima de cualquier base de datos
que "
?? "tenga el mismo nombre en el disco fijo."
?
?  "Después de ejecutar este programa, debe usar la opción Empaquetar y Rein-
dexar"
?? "para reconstruir las etiquetas de índices de todas las bases de datos an-
tes de "
?? "seguir trabajando con el sistema."
?
mconfirm = .F.
@ 12, 8 SAY "¿Está seguro de que desea continuar? (Y/N)" GET mconfirm;
        PICTURE "Y"
READ
DEACTIVATE WINDOW utility

IF mconfirm
   RUN RESTORE A: &msdrive.
   WAIT "Restauración del archivo terminada -- Pulse cualquier techa para con-
tinuar"
ENDIF

 lmargin = 0
 rmargin = 80
RELEASE WINDOW utility

RETURN

* Fin del programa NWRESTOR.PRG
```

Figura 27-9.

Programa para restaurar bases de datos a partir de copias de seguridad *(continuación)*.

de datos es necesario ejecutar la utilidad de reindexación inmediatamente después de restaurar las bases de datos en el disco fijo. En algunas aplicaciones, puede llamar al programa de reindexación automáticamente, antes de salir al programa de restauración, de modo que el operador no tenga que acordarse de cómo llevar a cabo este paso crucial.

CAPITULO VEINTIOCHO

Sistema de seguridad

La utilidad Protect de dBASE IV le permite establecer un sistema de seguridad protegido por palabra de paso que controla el acceso a dBASE IV, y opcionalmente, a las bases de datos y campos individuales dentro de las mismas. Es posible que esté acostumbrado a relacionar el concepto de seguridad con las redes, especialmente si tiene experiencia con minicomputadoras; pero aunque las redes y la seguridad suelen ir a menudo a la par, no son inseparables. La relación entre la red y la seguridad es histórica, ya que el software multiusuario ha emigrado a los microcomputadores desde los minis que soportan cientos o incluso miles de terminales. Cuando todos los archivos de datos y el software de la organización, que puede contener información altamente confidencial, se almacenan en la misma computadora, es esencial controlar el acceso de los usuarios a los programas y archivos de datos. El mismo tipo de consideraciones de seguridad también puede aplicarse a las redes de área local.

dBASE IV le permite proteger una instalación monousuario. Una computadora autónoma requiere un sistema de seguridad si algunos de los usuarios no deben trabajar con dBASE IV o si algunas aplicaciones que coexisten en el mismo disco fijo sólo deberían ser accesibles para algunos miembros del personal. En contraste, algunos pequeños negocios o departamentos de las compañías que utilizan redes de área local deben permitir que los usuarios compartan periféricos caros (tal como un disco fijo de alta capacidad o una impresora láser), o actualicen las mismas bases de datos de forma simultánea desde varias estaciones de trabajo, puede que necesiten restringir el acceso a cualquiera de los programas o datos.

Este capítulo describe cómo utilizar la utilidad Protect para definir una protección de palabra de paso proporcionando un nivel de seguridad adecuado a su organización. En este capítulo se resumen algunos de los métodos utilizados para controlar el acceso a los programas, y a las bases de datos de dBASE IV y se detallan técnicas de protección adicionales para las bases de datos independientes del sistema de seguridad incorporado en dBASE IV. El Capítulo 9 aplica estas técnicas a los sistemas de bases de datos multiusuario que se ejecutan en redes de área local.

USO DE LA UTILIDAD PROTECT

dBASE IV le permite implementar un sistema de seguridad protegido mediante una contraseña o palabra de paso a distintos niveles. Puede restringir el acceso al propio dBASE IV de modo que sea necesario introducir una palabra de paso para ejecutar el programa, y pueda ejercer un nivel más fino de control limitando el acceso a ciertas bases de datos o a ciertos campos individuales de una base de datos. La protección por palabra de paso se puede añadir en cualquier instante, en un sistema monousuario o en una instalación multiusuario, sin volver a configurar o instalar dBASE IV. Sin embargo, puesto que los programas existentes pueden requerir modificaciones para interactuar de forma inteligente con el sistema de seguridad, si sabe de antemano que una aplicación debe estar protegida, es mejor incorporar las características de seguridad en el diseño inicial.

Para controlar el acceso a dBASE IV, es necesario definir un perfil de identificación para cada usuario: el nombre de acceso, la contraseña, y el nombre del grupo. Una vez que ha definido el perfil del usuario, nadie podrá ejecutar dBASE IV sin introducir correctamente estos tres apartados. Para proteger una base de datos es necesario definir el perfil de acceso al archivo que identifica a los usuarios que pueden realizar operaciones, tales como adición, actualización, eliminación, y consulta de los datos. En el nivel más fino de control, puede definir el acceso a determinados campos estableciendo niveles de prioridad, es decir, puede permitir el acceso completo, hacer que un campo sea de sólo lectura, o prohibir el acceso completamente.

Cuando define un perfil de acceso a una base de datos, dBASE IV *cifra* o codifica el archivo .DBF, el correspondiente archivo .DBT (si la base de datos contiene campos memo) y cualquier archivo de índices asociado. Si utiliza la orden TYPE para visualizar un archivo cifrado, o lo examina con un editor de textos o una utilidad de inspección de disco (tales como la utilidad **Norton** o el programa DEBUG del MS-DOS), su contenido no será comprensible. Para visualizar los datos del archivo, dBASE IV debe *descifrar* el contenido.

Para definir y mantener el sistema de seguridad debe utilizar la utilidad Protect que está dirigida por menú. El sistema de seguridad se define desde el punto indicativo (o desde el Centro de Control), y no desde un programa de dBASE IV. Para acceder a la utilidad Protect desde el punto indicativo escriba

```
PROTECT
```

La utilidad Protect ha sido realizada basándose en la versión de dBASE III PLUS, y la apariencia de los menús y los mecanismos de selección de las opciones difieren poco de los menús de dBASE IV. Para seleccionar una opción, debe destacarla y pulsar ENTER. En respuesta, dBASE IV visualiza el área de datos, marcada con un triángulo pequeño, que se muestra en la Figura 28-1. Observe que si incluso se ha visualizado ya un valor, debe pulsar ENTER antes de que pueda editarlo o escribir encima de él.

La disposición de las opciones "almacenar" y "salir" también difieren algo de los menús de los editores de diseño de dBASE IV, que proporcionan una opción para almacenar su trabajo y salir: **Almacenar cambios y salir (Save changes and exit)**. En

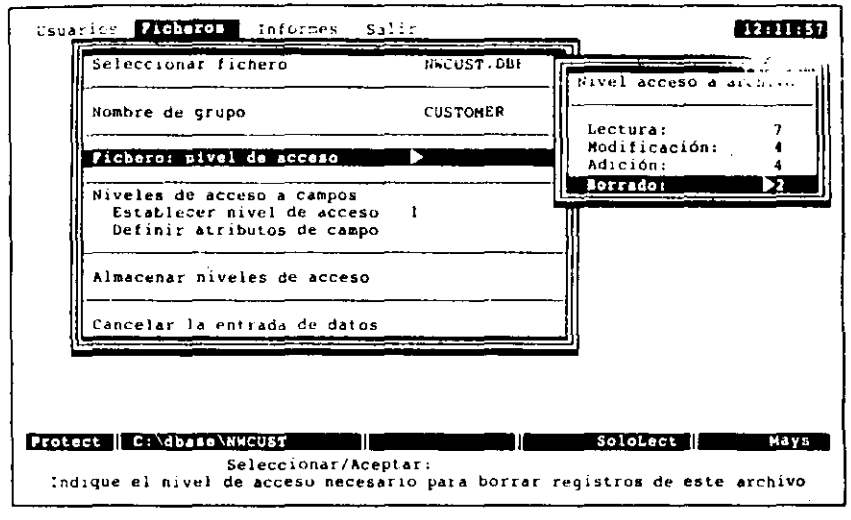

Figura 28-1.

Definición del nivel de acceso a los archivos.

el sistema Protect, siempre selecciona la opción **Almacenar (Save)** del menú **Salir (Exit)** para almacenar los perfiles de seguridad definidos en la sesión de trabajo actual. Después de seleccionar **Almacenar (Save)** el cursor avanza automáticamente a la opción **Salir (Exit)**; puede pulsar ENTER para salir o moverse a otro menú para continuar con su trabajo. Para salir sin almacenar sus cambios, debe seleccionar **Cancelar (Abandon)** del menú **Salir (Exit)** y luego seleccionar **Salir (Exit)**.

El administrador de dBASE IV

La persona encargada de manejar el sistema de seguridad se conoce como el *administrador de la base de datos*. El acceso a la utilidad Protect de dBASE IV está controlado por una contraseña especial llamada *palabra de paso del administrador*. En un negocio pequeño o en un grupo de trabajo, el título de administrador de la base de datos puede recaer en la persona que mejor comprenda el funcionamiento de dBASE IV; aunque puede que algunos usuarios conozcan la palabra de paso del administrador y utilicen la utilidad Protect para cifrar algunas bases de datos que creen. En una gran organización, el administrador de la base de datos puede ser el responsable de controlar el acceso a todas las computadoras de la compañía (incluyendo las minicomputadoras o maniframes), y esta persona puede ejercer un gran control sobre el sistema de seguridad de dBASE IV. Las discusiones de este capítulo suponen que usted, el programador, es el administrador de la base de datos o está trabajando jun-

to con él para diseñar el sistema de protección e integrar las medidas de seguridad necesarias en sus programas.

Cada vez que llama a la utilidad Protect, dBASE IV visualiza una ventana de diálogo para que introduzca la contraseña del administrador. Para proteger esta contraseña de observadores curiosos, los caracteres que escribe no se visualizan en la pantalla y, puesto que no ve lo que está escribiendo, debe verificar la nueva contraseña introduciéndola por segunda vez. La contraseña del administrador puede tener hasta 16 caracteres de longitud y puede contener cualquier carácter; cuando define la contraseña o palabra de paso, dBASE IV no hace distinción entre letras mayúsculas y minúsculas al igual que cuando la introduce más tarde para acceder de nuevo al sistema de seguridad.

dBASE IV almacena la contraseña del administrador en el disco de forma cifrada y una vez definida no hay forma de verla o modificarla. Cada organización debería asegurarse de que la contraseña del administrador está registrada y almacenada en un lugar seguro, de forma que en el supuesto de que el administrador de la base de datos la olvide o no esté localizable cuando urge conocerla pueda estar disponible.

Control de acceso a dBASE IV

Para controlar el acceso a dBASE IV, se utilizan las opciones del menú **Usuarios (Users)** para definir un perfil de identificación o acceso para cada usuario permitido. El perfil de identificación consta de cinco apartados: un nombre de acceso, la contraseña, un nombre de grupo, y opcionalmente el nombre completo y el nivel de acceso. Estos apartados se definen seleccionando las opciones apropiadas del menú **Usuarios (Users)**. Puede introducir los componentes del perfil de identificación con letras mayúsculas o minúsculas ya que no son significativas.

El *nombre de acceso* puede tener hasta 8 caracteres de longitud, y se utiliza para identificar a los usuarios. Es habitual, pero no necesario, asignar un único nombre de acceso, ya que no hay razón para hacer que los nombres de identificación estén cifrados o sean difíciles de recordar, dado que la contraseña por sí misma proporciona un alto nivel de seguridad. Algunas elecciones usuales para el nombre de acceso son una abreviatura del título de descripción del trabajo del usuario, el nombre o el apellido del usuario, sus iniciales, o su apodo.

La *contraseña* puede tener hasta 16 caracteres de longitud y siempre debería ser única para cada usuario, incluso si no está el nombre de acceso que el administrador de la base de datos sea el encargado de asignar las contraseñas o bien que sea el usuario quien la diga es una cuestión que depende del nivel de seguridad de su organización. En cualquier caso, la palabra de paso o contraseña debería ser difícil o imposible de adivinar, y como regla general no debería utilizar las primeras palabras que se le ocurran tales como apodos, nombre, apellidos, nombres de familiares, términos relacionados con el trabajo, nombres de cantantes favoritos, o cualquier otra palabra asociada con el individuo. Por otra parte, las contraseñas se componen de secuencias aleatorias de 16 letras y números que no necesariamente son difíciles de recordar. Una buena idea es utilizar dos o tres palabras castellanas no relacionadas, quizá separadas por signos de puntuación, que completen los 16 caracteres, por ejemplo "ojo-boca-nariz".

El *nombre de grupo*, que puede tener hasta 8 caracteres de longitud, se utiliza para relacionar los usuarios con los archivos de bases de datos, y en consecuencia, las aplicaciones, a las que necesitan acceder. Además asigna un nobre de grupo a un archivo de base de datos (utilizando las órdenes que se describen en el apartado siguiente), y dBASE IV sólo permite abrir el archivo a los usuarios con un nombre de grupo correspondiente. En el sistema National Widgets, puede asignar un nombre de grupo tal como NW, NWACTREC, o CUSTOMER para todos los archivos del sistema de cuentas por cobrar y usar el nombre de grupo MAILLIST para las bases de datos de listas para correspondencia por correo. Para trabajar con los archivos de más de un grupo, el usuario debe tener dos o más perfiles de identificación, cada uno de los cuales tiene un nombre de grupo diferente (el nombre de acceso y la contraseña puede ser el mismo). Si no asigna ninguna prioridad de acceso de archivo específico, el nombre de grupo sirve como uno de los tres elementos de la secuencia de identificación que valida el permiso del usuario para ejecutar dBASE IV.

Los 24 caracteres de la identificación o *nombre completo* representan una entrada opcional en el perfil de identificación del usuario. Puede introducir el nombre completo para identificar los individuos en los informes que documentan los sistemas de seguridad, o puede utilizarlo para registrar la asociación departamental de la persona. Por ejemplo, utilizaría la identificación completa para distinguir los miembros que pertenecen al centro de proceso de datos, al departamento de contabilidad, o al departamento de marketing o para observar un área principal de responsabilidad del usuario en la ejecución de una aplicación particular. dBASE IV no visualiza el nombre completo durante la secuencia de identificación o requiere que el usuario la introduzca; de hecho, puede observarla o imprimirla solamente desde dentro del sistema de seguridad.

Finalmente, cada perfil del usuario incluye *un nivel de acceso* que puede ir desde 1 hasta 8, el nivel con menor prioridad. Aunque este componente es necesario, por omisión se establece automáticamente en 1 (el nivel mayor); puede mantener este valor si no tiene que hacer distinción en el nivel de acceso de los usuarios. Junto con los niveles de acceso de archivo asignados a través del menú o mediante el menú **Archivos (Files)** descrito en la siguiente sección, el nivel de acceso determina qué archivos de bases de datos puede abrir un usuario y qué prioridades asigna dBASE IV al usuario para consultar o actualizar el archivo o determinados campos de archivo. Si sólo está implementando un sistema de seguridad para protegerse contra entradas no autorizadas en dBASE IV, debería mantener el nivel de acceso por omisión (1) para todos los usuarios.

Para almacenar el perfil de identificación, seleccione la opción **Almacenar datos usuario (Store user profile)** del menú **Usuarios (Users)**. Para consultar un usuario definido anteriormente, seleccione las opciones **Nombre acceso (Login name)**, **Contraseña (Password)** y **Grupo: nombre (Grup name)** para introducir los tres elementos del perfil de identificación que identifican de forma unívoca a un usuario. dBASE IV visualiza una ventana de diálogo para avisarle que el usuario ya está presente en el archivo, "El usuario ya existe, ¿desea editarlo? (S/N)" (User already exists, do you want to edit? (Y/N)). Si intenta volver a introducir inadvertidamente un usuario existente,

puede seleccionar NO del recuadro de diálogo. Si selecciona YES puede modificar el perfil del usuario existente o eliminar el usuario eligiendo **Borrar un usuario (Delete user from group)**.

dBASE IV almacena el perfil de los usuarios en un archivo llamado DBSYSTEM.DB, que está cifrado para evitar que un usuario experimentado examine este archivo para descubrir los nombres de identificación y las contraseñas. El esquema del cifrado basado en la contraseña del administrador, sin la cual no puede acceder a la pantalla Protect para ver o imprimir los perfiles de identificación.

Una vez que ha definido la contraseña del administrador y al menos un perfil de acceso del usuario, dBASE IV visualiza una pantalla de identificación, similar a la mostrada en la Figura 28-2, cuando cualquier usuario intenta arrancar el programa. dBASE IV acepta los tres elementos de identificación en letras mayúsculas o minúsculas (éste convierte automáticamente las entradas a mayúsculas), siendo difícil descubrir otro perfil de identificación de usuario, ya que dBASE IV no visualiza la palabra de paso en la pantalla. Si el usuario deja cualquiera de las entradas en blanco, dBASE IV sale inmediatamente; en cualquier otro caso, regresa al sistema operativo después de intentar tres veces sin éxito introducir un perfil de identificación no válido.

La orden LOGOUT cierra todas las bases abiertas e inicia de nuevo la secuencia de identificación sin salir de dBASE IV. En el punto indicativo, esta orden permite que un usuario ceda el control a otro o se identifique con un nombre de acceso o nombre de grupo diferente antes de arrancar otra aplicación o trabajar con otras bases de datos. Durante la comprobación de una aplicación protegida, puede identificarse como si fuera otro usuario diferente para verificar el rendimiento del control de acceso incorporado en sus programas. Aunque la orden LOGOUT está permitida en los programas de dBASE IV, ésta cierra todos los archivos de programa abiertos y le lleva al punto indicativo; no puede utilizar esta orden para permitir que un usuario se apodere de una estación de trabajo de otro mientras permanece en su aplicación.

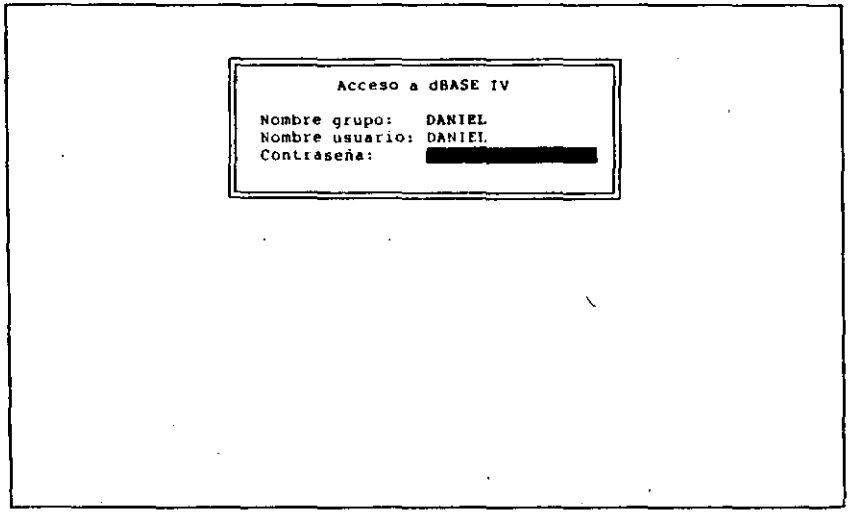

Figura 28-2.

La pantalla de identificación de dBASE IV.

Control de acceso a los archivos de bases de datos

Para controlar el acceso a determinados archivos de bases de datos y campos individuales de un archivo, utilice la opción del menú **Archivos (Files)** de la utilidad Protect para definir un perfil de acceso a cada una de las bases de datos protegidas. Al menos debe asignar un nombre de grupo, que dBASE IV compara con los nombres de grupo en los perfiles del usuario para determinar a quién le está permitido abrir el archivo para cualquier uso.

Puede definir un perfil de acceso más detallado estableciendo un nivel de acceso del usuario menor que le permita leer el archivo, actualizarlo (editar registros), ampliarlo (añadir nuevos registros) y marcar los registros. Recuerde que el nivel de acceso "menor" corresponde al número mayor de todos los posibles. Por ejemplo, en el sistema de National Widgets, puede asignar los siguientes niveles de acceso para el archivo Clientes:

Nivel de prioridad de lectura	7
Nivel de prioridad de actualización	4
Nivel de prioridad de ampliación	4
Nivel de prioridad de eliminación	2

De acuerdo con este esquema, los usuarios de cualquier nivel excepto los del nivel 8 pueden leer el archivo (visualizar datos); los niveles 1, 2, 3 y 4 permiten que un usuario actualice el archivo (edite registro, realice consultas de actualización, etc.) y amplíe el archivo (añada registros); y solamente los usuarios con niveles 1 y 2 pueden eliminar los registros. Por omisión, dBASE IV asigna a cada nuevo usuario un nivel de acceso 1 y establece los cuatro niveles de acceso en 8, de modo que los usuarios puedan acceder a todas las bases de datos. Si necesita separar a los usuarios y a los archivos en grupos de aplicaciones sin limitar las actividades de cualquier usuario, puede asignar a cada archivo un nombre de grupo y mantener los niveles de acceso por omisión.

Los niveles de acceso se definen seleccionando la opción **Archivo: nivel de acceso (File access privileges)** del menú **Archivos (Files)**, que llama a un recuadro de diálogo con cuatro opciones que corresponden a los cuatro tipos de prioridades de acceso a archivos. Aunque solamente puede asignar prioridades de archivo a nueve bases de datos simultáneamente, no tiene que salir y volver a entrar en la utilidad Protect antes de definir el décimo perfil de archivo: seleccione la opción del menú **Almacenar (Save)** del menú **Salir (Exit)** para almacenar el trabajo que ha hecho hasta ahora y

luego regrese al menú **Archivos (Files)** o **Usuarios (Users)** en lugar de elegir **Salir (Exit)**.

Cuando sale de la utilidad Protect después de almacenar al menos un perfil de usuario, dBASE IV actualiza el archivo DBSYSTEM.DB, que es el que almacena los perfiles de acceso del usuario. Además, hace una copia cifrada de cada una de las bases de datos a las que ha asignado un nombre de grupo (o un perfil de acceso más detallado). El archivo de base de datos cifrado tiene el mismo nombre que el original con la extensión .CRP. Si la base de datos contiene campos memo, dBASE IV también crea una copia cifrada del archivo .DBT con la extensión .CPT. El perfil de acceso al archivo se almacena en la base de datos cifrada, no en el archivo DBSYSTEM.DB. El cifrado de una base de datos extensa, puede tardar algún tiempo y debe disponer de suficiente espacio en su disco fijo para las copias cifradas y originales de los archivos .DBF y .DBT.

dBASE IV no cifra los índices .MDX ó .NDX, pero en las siguientes sesiones de trabajo cifrará los nuevos índices que incorpore a una base de datos protegida. Si reconstruye los índices existentes utilizando la orden REINDEX, dBASE IV cifra todas las etiquetas de índice .MDX y abre los índices .NDX. La forma más fácil de asegurar que todos los índices existentes para la nueva base de datos protegida estén cifrados es ejecutar el programa de reindexación y empaquetamiento escrito para su aplicación nada más salir de la utilidad Protect. Puesto que los índices puede que ya estén reconstruidos en base a la información de las bases de datos asociadas, dBASE IV no preserva el archivo .MDX o .NDX original cuando crea una copia cifrada.

Después de que dBASE IV cifra una base de datos, no elimina el archivo no protegido .DBF (o el archivo .DBT si la base de datos contiene campos memo) de su disco fijo. En la mayoría de los casos, deseará eliminar estos archivos y luego sustituir el archivo cifrado por el original renombrando los archivos .CRP y .CPT con las extensiones estándar .DBF y .DBT. Como precaución adicional, debería hacer copia de seguridad de los archivos que no están cifrados en discos flexibles o en cintas magnéticas antes de borrarlos; pero tanga en cuenta que los siguientes cambios que realice en la base de datos no estarán reflejados en sus copias de seguridad.

dBASE IV sólo limita el acceso a las bases de datos que están protegidas de forma explícita definiendo los perfiles de acceso. Otras bases de datos, así como los nuevos archivos definidos con la orden CREATE, no están cifrados y están disponibles para todos los usuarios. El estado del parámetro ENCRYPTION, que está en ON por omisión, determina si la base de datos cifra los nuevos archivos generados a partir de los archivos protegidos con órdenes tales como COPY, SORT y TOTAL. Cuando el parámetro ENCRYPTION está en ON, todas estas órdenes generan archivos cifrados y dBASE IV no le permite ejecutar órdenes que generan de forma invariable archivos no cifrados. Por ejemplo, para copiar o exportar datos a cualquier formato externo que no esté cifrado, debe asegurarse de que el parámetro ENCRYPTION esté en OFF.

Control del acceso a los campos de las bases de datos

Puede definir un perfil de acceso más detallado para una base de datos asignando prioridades de acceso a campos individuales para cada uno de los niveles de acceso

de usuario en general. Las prioridades permisibles son FULL, R/O (sólo lectura), y NONE. El valor por omisión en todos los casos es el acceso FULL, y debería definir las prioridades de forma explícita para cada uno de los ocho niveles de acceso que se asignan a cualquier usuario de su organización para anular este valor por omisión. Por ejemplo, si asigna las prioridades restringidas para niveles 2, 4 y 8, los usuarios en niveles de prioridad 1, 3, 5, 6 y 7 tendrían el acceso por omisión FULL para todos los campos.

Debe asignar niveles de prioridad de campo que sean consistentes con todos los niveles de acceso especificados para la base de datos. Por ejemplo, en el supuesto establecido en el apartado anterior para el archivo Clientes, para actualizar la base de datos en cualquier forma se requiere un nivel de acceso 4. Para los usuarios en los niveles 5, 6, 7 y 8, las prioridades de campo que tienen sentido son R/O y NONE. Si el archivo de las prioridades de campo está en conflicto, los privilegios de archivo tienen prioridad. Si ha concedido prioridades FULL a cualquier campo del archivo de Clientes para los usuarios con niveles de acceso 5, 6, 7 y 8, dBASE IV sólo permitirá que estos usuarios consulten los datos, cuando haya asignado el acceso R/O. La Tabla 28-1 lista las prioridades de acceso a los campos del archivo Clientes del sistema National Widgets.

Aun cuando defina un perfil de acceso para cada campo de una base de datos, todavía puede que necesite definir los cuatro niveles de prioridades que componen

Campo	Niveles 1,2	Niveles 3,4	Niveles 5,6,7	Nivel 8
ACCOUNT	FULL	FULL	R/O	NONE
COMPANY	FULL	FULL	R/O	NONE
ADDRESS1	FULL	FULL	R/O	NONE
ADDRESS2	FULL	FULL	R/O	NONE
CITY	FULL	FULL	R/O	NONE
STATE	FULL	FULL	R/O	NONE
ZIP	FULL	FULL	R/O	NONE
AREACODE	FULL	FULL	R/O	NONE
TELEPHONE	FULL	FULL	R/O	NONE
CONTACT	FULL	FULL	R/O	NONE
EQUIPMENT	FULL	FULL	R/O	NONE
FIRSTORDER	FULL	R/O	R/O	NONE
LASTORDER	FULL	R/O	R/O	NONE
CREDITOK	FULL	R/O	NONE	NONE
YTDINV	FULL	R/O	NONE	NONE
YTDPMT	FULL	R/O	NONE	NONE
TOTINV	FULL	R/O	NONE	NONE
TOTPMT	FULL	R/O	NONE	NONE
COMMENTS	FULL	FULL	R/O	NONE

Tabla 28-1.

Las prioridades de acceso a campos del archivo Clientes.

todos los perfiles de acceso al archivo para evitar que los usuarios no autorizados ejecuten órdenes que operen sobre la base de datos en general, tal como DELETE y ZAP. Sin embargo, si los usuarios no trabajan con bases de datos desde el punto indicativo, el perfil de archivo puede que no sea necesario ya que puede construir unas restricciones equivalentes en su aplicación.

Si regresa a la pantalla Protect para editar un nivel de acceso a archivo e intenta seleccionar un archivo protegido desde la lista de selección visualizada por la opción **Seleccionar archivo (New file)** del menú **Archivos (Files)**, dBASE IV visualiza el mensaje de error "La base de datos está cifrada" ("Database encrypted"). La única forma de revisar el nivel de acceso de una base de datos es generar una copia no cifrada del archivo, borrar la base de datos cifrada, volver a definir las prioridades de archivo, y permitir que dBASE IV cifre el archivo de nuevo. Si hace esto, debe volver a definir *todas* las prioridades de campo y archivo, ya que las especificaciones originales, que están almacenadas en la base de datos cifrada no están disponibles.

En las pantallas Hojear y Editar, dBASE IV sólo visualiza los campos para los cuales el usuario ha establecido los privilegios R/O o FULL, y utiliza los mismos colores o atributos de visualización monocromáticos para los campos R/O que para los mensajes de los nombres de campo (los colores especificados en una orden SET COLOR OF NORMAL), como si hubiera utilizado órdenes @ ... SAY en lugar de @ ... GET para visualizar los datos. Cualquier intento de cambiar un campo de sólo lectura (con órdenes tales como REPLACE) genera el mensaje de error "Nivel de acceso no permitido" ("Unauthorized access level"), y cualquier orden que incluya el nombre de un campo para el cual el permiso del usuario actual es NONE da por resultado el mensaje de error "Variable no encontrada" ("Variable not found").

Para los usuarios que trabajan desde el punto indicativo, el modo en que dBASE IV trata los campos de sólo lectura puede ser especialmente confuso, ya que los campos están visibles en la pantalla, aunque no están permitidas ciertas operaciones. Cuando el nivel de prioridad del usuario para un campo es NONE, dBASE IV se comporta de forma más consistente, es decir, como si el campo no existiera. Debe tener estos hechos en cuenta en sus programas ya que, como es de esperar, una orden @ ... GET que haga referencia a un campo para el cual la prioridad del usuario actual sea NONE genera el mensaje de error "Variable no encontrada" ("Variable not found"); pero otras órdenes que llevan a cabo acciones invisibles al usuario, tales como una orden STORE o una sentencia de asignación que inicialice una variable de memoria según un campo de la base de datos también hace eso. Cuando añada una protección a una aplicación de base de datos, puede que tenga que modificar todos sus archivos de formato y programas de entrada de datos existentes. Si sabe de antemano que una aplicación debe ser protegida, puede tener este hecho en cuenta en el diseño inicial.

En armonía con el esquema de prioridades de campos, cualquier array, base de datos, o archivos de texto que genere a partir de un archivo protegido contendrá solamente los campos para los que los usuarios conectados tienen prioridades R/O o FULL (sin tener en cuenta el estado del parámetro ENCRYPTION). Puesto que dBASE IV no copia los niveles de acceso a archivos en los nuevos generados con una orden COPY o SORT, un usuario que sepa cómo trabajar en el punto indicativo utilizaría la orden COPY para crear una nueva base de datos y editar los campos de esta base de datos que tengan prioridades R/O en la original. Sin embargo, este usuario no transferirá los cambios al archivo original.

Recuerde que puesto que dBASE IV almacena el nivel de acceso a una base de datos en el propio archivo de base de datos, no en el DBSYSTEM.DB, no puede definir varios niveles para grupos diferentes. Para conceder unos niveles de acceso altos a unos archivos de la aplicación y prioridades más restrictivas a otros, debe definir más de un perfil de identificación para el usuario; un perfil para cada aplicación con un nivel de acceso diferente (el nombre de acceso y la contraseña pueden ser idénticos).

PROTECCION DE UNA APLICACION DE dBASE IV

El sistema de seguridad establecido utilizando la utilidad Protect controla quién puede ejecutar dBASE IV y qué usuarios pueden trabajar con archivos de bases de datos individuales; pero no restringe el acceso a archivos de variables de memoria, formatos de etiquetas e informes, programas, y archivos de procedimientos. Sin embargo, en cualquier aplicación que requiera protección debe restringir el acceso a algunos programas y opciones de menú. En algunos sistemas, casi nadie debería tener permiso para imprimir informes, y sólo debería permitir actualizar las bases de datos a unos cuantos usuarios. En otras organizaciones, se cumple lo contrario: la entrada de datos la realizan muchos miembros del personal, pero los informes, que calculan y totalizan datos financieros confidenciales, sólo deben ser realizados por el controlador o el director de departamento.

Indirectamente, las prioridades de acceso a archivos determinan los programas que son accesibles, ya que cualquier programa que abra archivos de bases de datos generará eventualmente un mensaje de error si el nivel de acceso del usuario conectado prohíbe ejecutar cualquiera de las órdenes que contiene. Un dependiente encargado de los envíos de National Widgets que tenga permiso para introducir pedidos, pero no para cambiar cualquier campo del archivo de Clientes, no podría utilizar las opciones **Cambiar** o **Eliminar** del programa, pero incluso si modifica el programa de actualización para captar estos errores y visualizar mensajes más informativos que el mensaje "Variable no encontrada" o "Nivel de acceso no permitido" es mejor no permitir que el usuario empiece una operación prohibida en primer lugar. Además, puede que necesite restringir el acceso a las opciones del menú que no abren ningún archivo de base de datos, tales como la opción **Configuración del sistema** de algunas aplicaciones (aunque la versión descrita en el Capítulo 27 utiliza el archivo Códigos para validaciones).

De los cuatro componentes fundamentales de un perfil de identificación del usuario, nombre de grupo, nombre del usuario, contraseña y nivel de acceso, para un programa de dBASE IV solamente está disponible el nivel de acceso. La razón de esto es que una aplicación debería ser en la medida de lo posible independiente de los perfiles de identificación del usuario, que podrían cambiar en cualquier instante ya que el administrador de las bases de datos puede añadir nuevos usuarios, borrar usuarios que hayan dejado la compañía o el departamento, o cambiar periódicamente las contraseñas. El nivel de acceso, que podría estar de acuerdo con la política de la com-

pañía desde hace mucho tiempo, puede determinarse evaluando la función ACCESS. En un sistema monousuario, esta función da por resultado cero. Mientras el nivel de acceso es un determinante principal de las prioridades del usuario, es fácil controlar el acceso a programas individuales u opciones de menú comprobando el valor de esta función. Por ejemplo, podría modificar la estructura CASE del programa de menú de utilidades para permitir que los usuarios con un nivel de acceso 1 (el nivel más alto) ejecuten el programa de configuración del siguiente modo:

```
CASE BAR() = 8                    && Opciones de configuración del sistema
   IF ACCESS() <= 1
      DO NWsetup WITH .F.
   ELSE
      DO Message3 WITH "Esta opción no está disponible" + mspresskey
   ENDIF
```

En lugar de repetir estas órdenes muchas veces en todos los programas del menú, puede llevar a cabo la comprobación del nivel de acceso en un procedimiento tal como el siguiente:

```
PROCEDURE Acctest
PARAMETERS mprogram, maccess

IF ACCESS() <= maccess
   DO &mprogram.
ELSE
   DO Message3 WITH msnotavail" + mspresskey
ENDIF

RETURN
```

El procedimiento ACCTEST recibe dos entradas del programa que llama: MPROGRAM, que almacena el nombre del programa o procedimiento a ejecutar si el nivel de acceso del usuario lo permite, y MACCESS, que almacena el nivel de acceso inferior (el número más alto con permiso para ejecutar el programa especificado). El procedimiento comprueba el nivel de acceso del usuario determinado evaluando la función ACCESS, con MACCESS, y ejecuta el programa especificado (o procedimiento) o llama al procedimiento MESSAGE3 para visualizar un mensaje de error y detenerse hasta que el usuario pulsa una tecla. El mensaje "Esta opción no está disponible – Pulse cualquier tecla para continuar" se construye combinando dos variables de memoria públicas definidas en el programa NWSETUP.PRG: el texto se ha elegido de modo que sea más diplomático que el mensaje de error de dBASE IV "Nivel de acceso no permitido". La Figura 28-3 ilustra una nueva versión del programa del menú principal para el sistema National Widgets que llama al procedimiento ACCTEST. Este programa de menú nombra a varios programas de actualización (para los archivos de Transacciones e Inventario) que no se describen en este texto. Utilizando las técnicas descritas en el Capítulo 24, podría fácilmente escribir para estos archivos programas similares al programa de consulta y actualización de clientes. En el nuevo


```
* NWMMENU2.PRG
* ACCIONES DEL MENU PRINCIPAL
* ULTIMA REVISION: 07/18/89 M. LISKIN

ACTIVATE WINDOW menuscrn                  && Ventana para ejecutar la ac-
                                             ción del menú

CLEAR
DO CASE
   CASE BAR() = 3                         && Entrada del cliente
      DO Acctest WITH "NWcupd5", 5
   CASE BAR() = 4                         && Entrada del inventario
      DO Acctest WITH "NWinvupd", 4
   CASE BAR() = 5                         && Entrada del pedido
      DO Acctest WITH "NWordent", 8
   CASE BAR() = 6                         && Entrada de transacción
      DO Acctest WITH "NWtxnupd", 4
   CASE BAR() = 8                         && Etiqueta del menú de cartas
      ACTIVATE POPUP lblmenu
   CASE BAR() = 9                         && Menú de informe
      ACTIVATE POPUP rptmenu
   CASE BAR() = 11                        && Menú de utilidades
      DO Acctest WITH "NWumenu2", 8
   CASE BAR() = 13                        && Salir a dBASE IV
      DEACTIVATE WINDOW menuscrn
      DEACTIVATE POPUP
   CASE BAR() = 14                        && Salir al sistema operativo
      QUIT
ENDCASE

DEACTIVATE WINDOW menuscrn
RETURN

* Fin del programa NWMMENU2.PRG
```

Figura 28-3.
Una nueva versión del programa del menú principal.

programa de menú principal, cualquier usuario puede introducir pedidos, pero llamar al procedimiento ACCTEST le asegura que solamente los usuarios con niveles de acceso 5 o mayor (recuerde que un nivel de acceso mayor significa un número menor) puedan ejecutar el programa de actualización de clientes, y solamente los usuarios con niveles de acceso inferiores a 4 puedan actualizar el inventario y los datos de transacción.

En esta versión del programa, todas las opciones del menú llaman al procedimiento ACCTEST, incluso aquellas que especifican niveles de acceso inferiores, y permiten que cualquier usuario acceda a la opción. Esta estructura facilita posteriormente

el cambio del nivel de acceso requerido para ejecutar un programa cambiando el número en el procedimiento que llama. Si lo prefiere, puede utilizar simplemente una orden DO, tal como DO NWORDENT, para llamar a los programas que están disponibles para todos los usuarios. Además, puede sustituir las constantes que expresan los niveles de acceso en NWMMENU3.PRG por una serie de variables de memoria pública para que sea más fácil cambiar los permisos de toda la aplicación.

Si la orden DO ejecuta un programa que incluye parámetros, puede añadir la cláusula WITH que especifica los parámetros para la cadena de caracteres pasada a MPROGRAM cuando llama a ACCTEST. Estrictamente hablando, esta variable no almacena el nombre del programa, almacena cualquier cosa que siga al verbo "DO" en la orden que ejecuta el programa, que es ejecutado por el procedimiento ACCTEST en lugar del programa de menú. Por ejemplo, si aplicase este método de protección al menú de utilidades, podría escribir la llamada a ACCTEST que procesa la opción **Configuración del sistema** del siguiente modo:

```
DO Acctest WITH "NWsetup WITH .F.", 2
```

Además puede que necesite controlar el acceso a las opciones de menú que no llaman a órdenes DO, tal como las opciones que activan otros menús de aparición súbita o salen al punto indicativo u opciones para las que ha colocado la secuencia de órdenes requerida directamente en los programas de menú ya que fueron demasiado cortos para ser programas invocadores independientes. En el sistema National Widgets, los menús de etiquetas e informes están disponibles para todos los usuarios, y el menú de utilidades se define y se activa por otro programa y se ejecuta por una orden DO. Si necesita controlar el acceso al menú de informes, podría modificar las CASE correspondientes del programa del menú principal del siguiente modo:

```
CASE BAR() = 9                                  && Menú de informes
   IF ACCESS() <= 4
      ACTIVATE POPUP rptmenu
   ELSE
      DO Message3 WITH "Esta opción no está disponible" + msrpesskey
   ENDIF
```

Podría utilizar una técnica similar para prohibir a los usuarios con niveles de acceso inferiores al punto indicativo:

```
CASE BAR() = 13
   IF ACCESS() <= 4
      DEACTIVATE WINDOW menuscrn
      DEACTIVATE POPUP
   ELSE
      DO Message3 WITH "Esta opción está disponible" + mspresskey
   ENDIF
```

Recuerde que para evitar que un usuario trabaje en el punto indicativo debe asegurar que el usuario no pueda cargar dBASE IV sin arrancar su aplicación incluyendo una entrada COMMAND en el archivo CONFIG.DB que ejecute el programa de arranque principal. En el sistema National Widgets, podría utilizar la siguiente entrada en el CONFIG.DB:

```
COMMAND = DO NW2
```

Un método alternativo para proteger las opciones del menú es desactivar las líneas del menú de ventana o las opciones de la línea de menús de forma selectiva según el nivel de acceso de los usuarios conectados realmente empleando variantes de las órdenes DEFINE BAR, ON PAD, y ON SELECTION PAD. Para un menú de ventana, puede incluir una cláusula SKIP en cada orden DEFINE BAR como se ilustra en la nueva versión del programa del menú de utilidades mostrada en la Figura 28-4. Cuando ejecuta este programa, dBASE IV visualiza las opciones que no están disponibles para el usuario conectado actualmente en caracteres desvanecidos y no permite que el usuario seleccione estas opciones. Aunque ete método ofrece la ventaja de ser consistente con el sistema de menú de dBASE IV, a diferencia de la llamada explícita al procedimiento ACCTEST, no proporciona al usuario una explicación sobre el porqué ciertas opciones están restringidas.

En el sistema National Widgets, sólo los usuarios con niveles de acceso 4 o superior pueden ejecutar órdenes del DOS, y aunque todos los usuarios pueden hacer copias de seguridad de las bases de datos, solamente los usuarios con niveles de acceso 1 ó 2 pueden restaurar las bases de datos en el disco fijo desde copias de seguridad. La razón de esto es que el programa de restauración no debería ser accesible a usuarios que no comprendan claramente las consecuencias de reemplazar la información actual con datos obsoletos almacenados en los discos que contienen las copias de seguridad si la opción se elige en un momento no apropiado. Para ejecutar el programa


```
* NWUMENU.PRG
* ACCIONES DEL MENU DE UTILIDADES
* ULTIMA REVISION: 07/18/89 M. LISKIN

DO Menudef

ACTIVATE WINDOW utilmenu
ACTIVATE POPUP utilmenu

DEACTIVATE WINDOW utilmenu
RETURN

*************************************************************************

* Define las ventanas y el menú
PROCEDURE Menudef
```

Figura 28-4.

Una nueva versión del programa del menú de utilidades.

```
* Ventana para visualizar el menú
DEFINE WINDOW utilmenu FROM 0, 0 TO 24,79 NONE

* Opciones del menú de utilidades

DEFINE POPUP utilmenu FROM 4,21 TO 21,55 ;
   MESSAGE "Seleccione una opción, destáquela y pulse <ENTER> " +;
   "o pulse la primera letra".
DEFINE BAR  1 OF utilmenu PROMPT     "Menú de utilidades"      SKIP
DEFINE BAR  2 OF utilmenu PROMPT "------------------------" SKIP
DEFINE BAR  3 OF utilmenu PROMPT "Comprobar el espacio en disco";
DEFINE BAR  4 OF utilmenu PROMPT "Formatear discos flexibles";
DEFINE BAR  5 OF utilmenu PROMPT "Ejecutar órdenes DOS";
DEFINE BAR  6 OF utilmenu PROMPT "Copias de seguridad de bases de datos";
DEFINE BAR  7 OF utilmenu PROMPT "Restaurar copias de seguridad";
   SKIP FOR ACCESS() > 2
DEFINE BAR  8 OF utilmenu PROMPT "Establecer opciones del sistema";
   SKIP FOR ACCESS() > 1
DEFINE BAR  9 OF utilmenu PROMPT "-------------------------"; SKIP
DEFINE BAR 10 OF utilmenu PROMPT "Empaquetar y/o Reindexar bases de datos";
DEFINE BAR 11 OF utilmenu PROMPT "Exportar datos del cliente";
DEFINE BAR 12 OF utilmenu PROMPT "Validar el archivo de clientes";
DEFINE BAR 13 OF utilmenu PROMPT "-------------------------"; SKIP
DEFINE BAR 14 OF utilmenu PROMPT "Pone a cero campos YTD de clientes";
   SKIP FOR ACCESS() > 1
DEFINE BAR 15 OF utilmenu PROMPT "Reconstruir los totales del cliente";
   SKIP FOR ACCESS() > 1
DEFINE BAR 16 OF utilmenu PROMPT "Depurar archivos por fecha";
   SKIP FOR ACCESS() > 1
DEFINE BAR 17 OF utilmenu PROMPT "Cambiar códigos de cuenta de clientes";
   SKIP FOR ACCESS() > 1
ON SELECTION POPUP utilmenu DO Utilmenu

RETURN

* Acciones del menú utilidades
PROCEDURE Utilmenu

DO CASE
   CAS BAR() = 3                          && Comprobar el espacio en disco
      CLEAR
      RUN CHKDSK
   CASE BAR() = 4                         && Formatear discos
      RUN FORMAT A:
   CASE BAR() = 5                         && Ejecutar órdenes del DOS
      DO NWrundos
```

Figura 28-4.

Una nueva versión del programa del menú de utilidades *(continuación)*.


```
   CASE BAR() = 6                         && Copias de seguridad de bases
                                             de datos
      USE NW.CAT ALIAS Filelist           && Abre el catálogo como una base
                                             de datos
      DO NWbackup
   CASE BAR() = 7                         && Retaurar bases de datos
      DO NWRestor
   CASE BAR() = 8                         && Establecer las opciones del
                                             sistema
      DO NWsetup WITH .F.
   CASE BAR() = 10                        && Empaquetar y/o reindexar bases
                                             de datos
      USE NW.CAT ALIAS Filelist           && Abre el catálogo como una base
de datos
      DO NWindex4
   CASE BAR() = 11                        && Exportar datos del cliente
      DO NWcopy
   CASE BAR() = 12                        && Validar el archivo Clientes
      DO NWcvalid
   CASE BAR() = 14                        && Pone a cero los campos YTD del
                                             cliente
      DO NWczero
   CASE BAR() = 15                        && Reconstruir los totales del
                                             cliente
      DO NWtotal
   CASE BAR() = 16                        && Depurar los archivos por fecha
      DO NWpurge
   CASE BAR() = 17                        && Cambiar el código de cuenta
                                             del cliente
      DO NWchgact
ENDCASE

RETURN

* Fin del procedimiento Utilmenu

* Fin del programa NWUMENU.PRG
```

Figura 28-4.

Una nueva versión del programa del menú de utilidades *(continuación)*.

de configuración del sistema se requiere un nivel de acceso 1 y las cuatro últimas opciones de mantenimiento de la base de datos que modifican y depuran los mismos.

Puede utilizar técnicas similares para controlar el acceso a las opciones individuales desde un programa. Por ejemplo, en el programa de consulta y actualización de clientes de National Widgets (que es accesible a usuarios con niveles de acceso 5 o superior, es decir, números de nivel de acceso 1, 2, 3, 4 y 5), puede requerir un nivel de acceso 1 ó 2 para consultar los pedidos a las transacciones de actualización y per-

mitir que solamente marquen los clientes los usuarios con niveles de acceso 1. Además, puesto que los usuarios con nivel de acceso 5 no tienen permiso FULL para los campos de archivo de Clientes, no deberían tener permiso para seleccionar la opción **Introducir** o **Cambiar**. Para implementar este esquema de protección, puede sustituir la llamada al procedimiento ACCTEST por órdenes DO en las órdenes ON SELECTION PAD que definen las correspondientes opciones del menú, del siguiente modo:

```
ON SELECTION PAD Enter OF custupd DO Acctest WITH "Custupd", 4
ON SELECTION PAD Change OF custupd DO Acctest WITH "Custupd", 4
ON SELECTION PAD Delete OF custupd DO Acctest WITH "Custupd", 1
ON SELECTION PAD Transact OF custupd DO Acctest WITH "Custupd", 2
ON SELECTION PAD Orders OF custupd DO Acctest WITH "Custupd", 2
```

Otra forma de desactivar la línea de menús de forma selectiva según el nivel de acceso del usuario, es ejecutar una orden ON SELECTION PAD sin ninguna acción (o una orden ON PAD si la opción de la línea de menús llama a un menú desplegable). En el programa de actualización de clientes, podría colocar después de la serie de órdenes ON SELECTION PAD que definen la línea de menús principal las siguientes órdenes:

```
IF ACCESS() > 4
   ON SELECTION PAD Enter OF custupd
   ON SELECTION PAD Change OF custupd
ENDIF
IF ACCESS() > 1
   ON SELECTION PAD Delete OF custupd
ENDIF
IF ACCESS() > 2
   ON SELECTION PAD Transact OF custupd
   ON SELECTION PAD Orders OF custupd
ENDIF
```

Este método es menos directo que la adición de una cláusula SKIP a la orden DEFINE BAR; pero utiliza solamente órdenes que son propias de dBASE IV. Sin embargo, dBASE IV no identifica visualmente las opciones que no están disponibles visualizándolas de forma poco clara o en otro color diferente; simplemente ignora cualquier intento de seleccionar la opción especificada.

Estos métodos son útiles para definir una rutina de captación de errores que detecte y responda al mensaje de error de dBASE IV estándar activado cuando el usuario intenta realizar una operación prohibida. Por ejemplo, el nivel de prioridad de archivo establecido para el archivo de Clientes prohibe que los usuarios con niveles de acceso inferior a 2 eliminen los registros, pero la condición de error ocurre solamente cuando dBASE IV intenta ejecutar una orden DELETE. Si permite que un usuario con niveles de acceso 3, 4 ó 5 seleccione la opción **Eliminar** del programa de actualización de clientes, el programa comprobaría el campo de balance, buscaría las transacciones coincidentes, y si el registro supera estas dos condiciones, pediría que el usuario confirme la operación antes de ejecutar la orden DELETE. Verificando la función ACCESS cuando el usuario selecciona la opción de menú por primera vez, puede evitar que el usuario avance a través de este callejón sin salida.

El sistema de seguridad resumido para el programa de consulta y actualización de clientes presenta la ventaja de proporcionar más niveles de acceso de los dos o tres requeridos generalmente por la mayoría de las aplicaciones. Con los ocho niveles a su disposición, puede asignar prioridades similares a varios niveles de acceso, para diferenciarlos en sus programas. Por ejemplo, los usuarios con niveles de acceso 1 ó 2 tienen la misma prioridad de acceso a los campos del archivo de Clientes; pero el programa de actualización no permite que los usuarios con niveles de acceso 2 los marquen para su eliminación.

Control de acceso a los archivos y campos de los programas

Como se dijo anteriormente en este capítulo, dBASE IV responde con un mensaje de error a cualquier intento de ejecutar una orden que viole las prioridades establecidas para los usuarios conectados actualmente, dependiendo del perfil del usuario y el nivel de acceso al archivo. A nivel de mandato, dBASE IV visualiza los errores estándar en una ventana de diálogo, que ofrece normalmente las elecciones: **Cancelar, Editar** y **Ayuda** (Cancel, Edit y Help). En una aplicación dirigida por menú, puede en general evitar los mensajes de error de dBASE IV utilizando un programa o procedimiento de captación de errores llamado con una orden ON ERROR, pero en un programa interactivo altamente completo tal como el programa de consulta y actualización de clientes, no debe confiar en la comprobación de nivel de acceso automática que dBASE IV realiza cuando visualiza y capta campos con órdenes @... GET.

Cuando ejecuta los programas de actualización y consulta de clientes escrito en el Capítulo 24, dBASE IV visualiza los campos para los cuales los usuarios actuales tienen un permiso R/O en los colores utilizados para los datos visualizados con @... SAY (aunque el programa utiliza órdenes @... GET para visualizar todos los campos). Al igual que en las pantallas Editar y Hojear, el usuario puede desplazar el cursor a través de ellos, pero no puede hacer modificaciones. Aunque este comportamiento puede ser algo confuso, crea problemas cuando el contenido presente en un campo no supere la condición especificada en la cláusula RANGE o VALID y el usuario no puede editar los datos para hacer una corrección (como se dijo en el Capítulo 7, dBASE IV aplica estas condiciones al valor de campos existentes así como a las nuevas entradas de datos). Estas propiedades de las cláusulas RANGE y VALID son problemáticas, incluso cuando no tiene el sistema de seguridad activo y la mejor solución es evitar el uso de las pruebas de validación que no superan los registros existentes.

El mensaje de error "Variable no encontrada" ("Variable not found") que ocurre cuando cualquier orden nombra a un campo para el cual la prioridad de acceso del usuario actual es NONE presenta un problema más serio. La sentencia de asignación que almacena el contenido del campo CITY del último registro introducido en la variable de memoria MCITY (que proporciona el valor por omisión para el siguiente cliente) generará este mensaje de error si el nivel de acceso del usuario actual es 5, aun cuando este usuario no pueda seleccionar la opción **Introducir** para añadir un nue-

vo cliente. Podría utilizar en la aplicación un procedimiento de tratamiento de errores que ejecute una orden RETURN para ignorar el error y regresar al programa que llama. No obstante, las órdenes @... GET del procedimiento CUSTGETS que visualizan los campos prohibidos generarán el mismo mensaje de error, en este caso, el programa no ignoraría el error. En su lugar, puede volver a estructurar el programa de modo que todas las órdenes que actúan sobre los campos de la base de datos dependan del valor de la función ACCESS. Por ejemplo, podría inicializar la variable MCITY en la siguiente estructura IF:

```
IF ACCESS() <= 4
   GOTO RECCOUNT()
   mcity = City
ENDIF
```

Para evitar este problema no es posible evitar una cláusula WHEN que compruebe la función ACCESS, ya que el error se activa por la referencia inicial al nombre de la variable en la cláusula GET antes de que dBASE IV procese la condición de la cláusula WHEN. En su lugar, debe encerrar las órdenes @ ... SAY ... GET que visualizan o captan los campos en una estructura IF basada en el valor de la función ACCESS.

La Figura 28-5 ilustra una nueva versión del procedimiento CUSTGETS que pone de manifiesto esta técnica para los campos de totales financieros, que no son accesibles a usuarios cuyos niveles de acceso sea menor o igual a 5 (números de nivel de acceso de 5 a 8). No hay razón para recordar a los usuarios la existencia de campos que no pueden consultarse, de modo que el procedimiento CUSTSAYS debería modificarse mientras las líneas que componen las órdenes @ ... SAY que visualizan el texto asociado al campo sean contingentes con los niveles de acceso del usuario. La omisión de todos los campos financieros deja la mitad de la pantalla inferior vacía, de modo que el nuevo procedimiento CUSTGETS abre una ventana llamada COMMENTS2 para visualizar el contenido del campo memo. El procedimiento CUSTPRINT incluye una estructura IF para que el usuario que no tiene permiso para visualizar los campos financieros en la pantalla no pueda imprimirlos.

Una vez que incorpore todas las condiciones de niveles de acceso descritas en este capítulo en un programa de consulta y actualización de bases de datos, el programa no dependerá de la forma en que se llevan a cabo las condiciones de nivel de acceso incorporadas en dBASE IV siempre que el sistema de seguridad esté activo. Si en los programas escribe explícitamente todas estas condiciones de acceso, puede considerar que concede a todos los usuarios un acceso FULL a los campos. Haciendo esto, puede eliminar las estructuras IF, tal como aquella que impide la inicialización de MCITY que genera un mensaje de error. No obstante, puede que mantener los perfiles de acceso a campos establecidos con la utilidad Protect si los usuarios no tienen permiso para consultar o actualizar datos fuera de su aplicación, desde el Centro de Control o desde el punto indicativo.


```
PROCEDURE Custgets

@ 0,10 SAY IIF(DELETED(), "*Marcado*", SPACE(11))
@ 2,13 GET Account
CLEAR GETS
@ 4,13 GET Company
@ 5,13 GET Address1 VALID ADDRESS1 <> " ";
       ERROR "La línea de la primera dirección no debe estar en blanco"
@ 6,13 GET Address2
@ 7,13 GET City VALID CITY <> " ";
       ERROR "La ciudad no debe estar en blanco"
@ 7,52 GET State PICTURE "@A!" VALID SEEK("S"+State, "Codes");
       ERROR "El estado debe ser una abreviatura de dos letras"
@ 7,62 GET Zip PICTURE "@A!" VALID Zipok(Zip, " ") WHEN State <> " "
@ 4,53 GET Areacode PICTURE "999" VALID SEEK("A"+Areacode, "Codes");
       ERROR "No es un código de área válido"
@ 4,59 GET Telephone PICTURE "999-9999"
@ 6,52 GET Contact VALID Contact <> " " .OR. Company <> " ";
       ERROR "El nombre de la compañía y del contacto no pueden estar en blanco a la vez"
@ 9,13 GET Equipment PICTURE "@S62"
IF ACCESS() <= 4
   @ 11, 8 GET Comments OPEN WINDOW comment1
ELSE
   @ 11, 8 GET Comments OPEN WINDOW comment2
ENDIF
IF ACCESS() <= 4
   @ 17,15 GET Firstorder WHEN Firstorder = { / / }
   @ 18,15 GET Lastorder ;
           VALID Lastorder >= Firstorder .AND. Lastorder <= DATE();
           ERROR "El último pedido no puede ser anterior al primero o posterior a la fecha actual";
           MESSAGE "Use la fecha de la factura, no la feche del cheque de los pedidos por anticipado"
   @ 17,68 GET Totinv PICTURE "99,999.99"
   @ 17,39 GET Ytdinv PICTURE "99,999.99"
   @ 18,39 GET Ytdpmt PICTURE "99,999.99"
   @ 18,68 GET Totpmt PICTURE "99,999.99"
   @ 20,15 GET Creditok PICTURE "Y"
   @ 20,67 SAY Totinv - Totpmt PICTURE "@$ 999,999.99"
ENDIF
* Vuelve a visualizar el estado y el código de área
DO Descrip

RETURN
* Fin del procedimiento Custgets
```

Figura 28-5.

Un procedimiento que capta campos de forma condicional según el nivel de acceso del usuario.

	Niveles de Acceso							
Nombre	1	2	3	4	5	6	7	8
ACCOUNT	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
COMPANY	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
ADDRESS1	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
ADDRESS2	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
CITY	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
STATE	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
ZIP	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
AREACODE	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
TELEPHONE	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
CONTACT	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
EQUIPMENT	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE
FIRSTORDER	FULL	FULL	R/O	R/O	R/O	R/O	R/O	NONE
LASTORDER	FULL	FULL	R/O	R/O	R/O	R/O	R/O	NONE
CREDITOK	FULL	FULL	R/O	R/O	NONE	NONE	NONE	NONE
YTDINV	FULL	FULL	R/O	R/O	NONE	NONE	NONE	NONE
YTDPMT	FULL	FULL	R/O	R/O	NONE	NONE	NONE	NONE
TOTINV	FULL	FULL	R/O	R/O	NONE	NONE	NONE	NONE
TOTPMT	FULL	FULL	R/O	R/O	NONE	NONE	NONE	NONE
COMMENTS	FULL	FULL	FULL	FULL	R/O	R/O	R/O	NONE

Figura 28-7.

Informe de archivos impreso por la utilidad Protect.

CAPITULO VEINTINUEVE

Escritura de aplicaciones multiusuario para redes

El Capítulo 28 describió cómo diseñar e implementar un sistema de seguridad protegido por contraseña o palabra de paso para controlar el acceso a dBASE IV y a los programas y archivos que componen una aplicación de dBASE IV. Con frecuencia se requiere un sistema de seguridad en una aplicación multiusuario diseñada para ejecutarse en una red de área local, pero cuando ejecuta dBASE IV (o cualquier otro software) en una red, es de una gran importancia otro tipo de seguridad: debe proteger los archivos de los daños que podrían producirse si dos usuarios intentan actualizar los mismos datos simultáneamente. Cuando un usuario está realizando una operación que implica actualizaciones de una base de datos a gran escala, el archivo completo debe estar cerrado para otros usuarios. Cuando varios usuarios editan registros uno a uno en la pantalla Editar u Hojear, o mediante un programa de actualización, tal como el descrito en el Capítulo 24, es necesario asegurar que no puedan editar el mismo registro simultáneamente.

Este capítulo resume los conflictos potenciales que pueden surgir en una aplicación de red y describe las estrategias a seguir para resolver estos problemas. Además, presenta nuevas versiones de algunos programas del sistema National Widgets, que incluyen un programa de actualización multiusuario para el archivo Clientes. Debido a las muchas diferencias existentes entre las redes de área local, soportadas por dBASE IV y las distintas configuraciones de redes que pueden instalarse en su organización, este libro no describe cómo configurar su software de sistema operativo de red o cómo instalar dBASE IV en el servidor de archivos. Este capítulo supone que el software ya está instalado y que sabe cómo arrancar el programa e imprimir desde su estación de trabajo.

contenido de un archivo. Por ejemplo, dBASE IV abre un archivo de formato de pantalla, informe, o etiqueta en modo exclusivo cuando llama al editor de diseño apropiado para modificar el formato, y en modo compartido cuando utiliza el formato solamente para visualizar o imprimir los datos. Los archivos de índices .MDX y .NDX tienen siempre asignado el mismo modo así como los archivos de base de datos asociados.

Es posible anular el valor por omisión del modo de apertura de un archivo y requerir el acceso exclusivo para todos los archivos estableciendo el parámetro EXCLUSIVE en ON (por omisión, este parámetro está en OFF para permitir el acceso simultáneo). También puede abrir cualquier archivo de base de datos para uso exclusivo, independientemente del estado del parámetro EXCLUSIVE, incluyendo la palabra clave EXCLUSIVE en la orden USE. Por ejemplo, podría abrir el archivo Clientes de National Widgets (y el archivo de producción .MDX) en modo exclusivo con una orden tal como la siguiente:

```
USE NWcust ORDER Account ALIAS Customer EXCLUSIVE
```

El cambio de estado del parámetro EXCLUSIVE no tiene efecto sobre los archivos que ya están abiertos. Si una base de datos está abierta en modo compartido, debe cerrar el archivo y volverlo a abrir para pasar al modo exclusivo.

Para llevar a cabo cualquier orden que vuelva a escribir el archivo completamente, incluyendo PACK, ZAP, INDEX o REINDEX *debe* abrir una base de datos en modo exclusivo. Un intento de ejecutar una de estas órdenes desde un programa de dBASE IV en una base de datos que ha sido abierta en modo compartido daría por resultado el mensaje de error "El archivo debe abrirse en modo exclusivo" ("File must be opened in exclusive mode"). Si intenta abrir un archivo en modo exclusivo cuando el archivo ya está abierto en otra estación de trabajo, dBASE IV genera el mensaje de error "El archivo está en uso" ("File in use by another") es suficiente para segurar que los usuarios que se encuentran en estaciones de trabajo físicamente distantes no abrirán de forma inadvertida el mismo archivo cuando tendrían la posibilidad de dañar los datos si lo hacen. Sus programas de dBASE IV deben tratar las dos posibles condiciones de error que pueden surgir si hay necesidad de abrir archivos en modo exclusivo visualizando mensajes informativos y, en la mayoría de los casos, permitiendo que el usuario decida qué hacer a continuación.

Por ejemplo, antes de ejecutar el sistema National Widgets en una red, tendría que modificar el programa de reindexar y empaquetar de modo que abra cada una de las bases de datos en modo exclusivo, bien añadiendo la palabra clave EXCLUSIVE a la orden USE o colocando una orden SET EXCLUSIVE ON al principio del programa y SET EXCLUSIVE OFF al final. El sistema debe tratar las condiciones de error que surgen si una de las bases de datos no puede abrirse para uso exclusivo porque ya está abierta en otra estación de trabajo. Además, también debe captar el error generado en otra estación de trabajo si un segundo usuario intenta abrir la base de datos que el programa que reindexa y empaqueta actualmente tiene abierta en modo exclusivo.

Puede tratar ambas contingencias en un programa o procedimiento de captación de errores, tal como el programa NWERROR.PRG descrito en el Capítulo 25. La

aproximación más simple es volver desde la rutina de captación de errores con RETRY en lugar de con RETURN:

```
CASE ERROR() = 108 .OR. ERROR() = 372
   RETRY
```

Al igual que RETURN, la orden RETRY hace que el programa o procedimiento actual determine y regrese al programa o procedimiento que llama; pero RETRY hace que dBASE IV repita la orden del programa invocador que generó el error en lugar de proceder con la orden siguiente. Esta aproximación mínima pasa normalmente inadvertida. Si el usuario que ha abierto originalmente el archivo está realizando una operación de larga duración, debería dar a otros usuarios la opción de que esperen para intentar de nuevo más tarde el acceso al archivo en lugar de que sigan intentándolo desde sus estaciones de trabajo mientras esperan que el archivo esté disponible. Además, si dos usuarios empiezan a ejecutar programas que abren las mismas bases de datos para uso exclusivo, pero en orden opuesto, puede dar por resultado un *interbloqueo* que a menudo se denomina "abrazo mortal".

Por ejemplo, si un usuario abre el archivo Clientes en modo exclusivo y otro abre el archivo Pedidos para uso exclusivo al mismo tiempo, ningún usuario será capaz de acceder al otro archivo. Cuando el primer usuario intenta abrir el archivo Pedidos, éste ya está abierto en modo exclusivo en la segunda estación de trabajo, y cuando el segundo usuario intenta abrir el archivo Clientes, éste estará abierto en modo exclusivo en la primera. dBASE IV no puede resolver este conflicto, y si su rutina de captación de errores le devuelve al programa que llama con una orden RETRY, los programas que se ejecutan en ambas estaciones de trabajo continuarán intentando el proceso de forma indefinida y el sistema sufrirá una parada imprevista.

Una solución a este problema es esperar un intervalo de tiempo determinado, ejecutar la orden RETRY y, si el archivo no está todavía disponible, visualizar un mensaje informativo al operador y ofrecer la advertencia "Inténtelo de nuevo más tarde". El procedimiento de captación de errores debe incluir un bucle que cuente hasta 100 (o algún otro número) antes de ejecutar la orden RETRY. Después de la primera RETRY, si el segundo intento de ejecutar la orden en el programa que llama da el mismo error, el programa de captación de errores no debería contar hasta 100 y salir sin RETRY. El procedimiento debería ejecutar una orden RETURN y permitir que el programa invocador determine cómo tratar la situación. La siguiente rutina de recuperación de errores ilustra este hecho:

```
CASE ERROR() = 108 .OR. ERROR() = 372        && Archivo en uso por otro
   mcount = 1
      DO WHILE mcount <= 100
         mcount = mcount + 1
      ENDDO
   IF mretry
      mretry = .F.
      RETRY
   ELSE
```

```
    merror = .T.
    DO Message3 WITH "La base de datos solicitada está en uso en otra " +;
       "estación de trabajo. Por favor inténtelo de nuevo más tarde."
    RETURN
ENDIF
```

Esta rutina se comunica con el programa que llama por medio de dos variables de memoria: MRETRY y MERROR. La variable MRETRY determina si la rutina de captación de errores regresa al programa que llama con RETRY o visualiza un mensaje de error y sale con RETURN. Esta variable debe ser inicializada en el programa que llama con el valor .T. para indicar que el manipulador de error saldría con una orden RETRY. MRETRY está establecida en .F. por la rutina de tratamiento de errores antes de ejecutar la orden RETRY, de modo que si la recuperación da lugar al mismo mensaje de error, el procedimiento visualiza el mensaje y sale con RETURN la siguiente vez que es llamado. No puede elegir entre RETRY y RETURN en función del valor de la función ERROR, ya que ambas órdenes ponen a cero el valor devuelto por esta función. La única forma de mantener la información sobre la condición de error anterior, en este caso, el resultado del primer intento de abrir una base de datos, es almacenarla en una variable de memoria.

Si no logra ejecutar la orden original al segundo intento, el manipulador de error de propósito general no puede determinar qué hacer a continuación: la decisión debe ser tomada por el programa que llama. Si no existe otro recurso y sale al menú principal cuando es imposible abrir un archivo, debe regresar al programa invocador para "limpiar" el entorno: cerrar cualquier base de datos que esté abierta, desactivar los menús y las ventanas y, si existen menús, ventanas o variables públicas definidas dentro del programa, borrar su definición de la memoria. En algunos casos, puede proporcionar al usuario la opción de volver a intentarlo dos o tres veces o incluso indefinidamente, antes de regresar al menú. La variable de memoria MERROR permite que el programa que llama determine si la condición de error se ha superado con éxito o no con RETRY. Esta variable de memoria se inicializa con el valor .F. en el programa llamado y se vuelve a poner a .T. si la rutina de captación de errores determina que existe todavía la condición de error y que la orden original no se ha ejecutado con éxito.

En lugar de utilizar un bucle DO WHILE para hacer una pausa, puede aprovecharse del parámetro REPROCESS, que acepta un valor numérico que especifica el número de veces que dBASE IV intenta realizar cualquier operación de red que genera un error antes de iniciar la secuencia de respuesta de error estándar. Por ejemplo, para hacer que dBASE IV vuelva a intentar la ejecución de cualquier orden que genera el error de red 10 veces, utilizaría la siguiente orden (colocada en el programa de menú principal, la rutina de captación de errores, o en un programa que pueda generar un error de red):

```
SET REPROCESS TO 10
```

Luego volvería a escribir la rutina de captación de errores del siguiente modo:


```
CASE ERROR() = 108 .OR. ERROR() = 372          && El archivo está en uso en otra
                                                  estación
   merror = .T.
   DO Message3 WITH "La base de datos solicitada está en uso en otra " +;
      "estación de trabajo. Inténtelo de nuevo más tarde."
   RETURN
```

Puesto que el parámetro REPROCESS hace que dBASE IV vuelva a intentar la ejecución de la orden que genera el error 10 veces antes de pasar el control al procedimiento de captación de errores no necesita la variable MRETRY (si uno de los diez intentos se realiza con éxito, la rutina de captación de errores no será ejecutada nunca).

Si desea que el usuario decida las veces que intentará la ejecución de una orden que genera el mensaje de error "El archivo está en uso" ("File in use by another"), puede modificar la rutina de captación de errores de modo que visualice más información para que le ayude a tomar la decisión más apropiada y luego permita que el usuario decida si vuelve a ejecutar la orden o no. La siguiente versión de la rutina de captación de errores, que asume la orden SET REPROCESS anterior, solicita del usuario si vuelve a intentar abrir un archivo, y según la respuesta, ejecuta una orden RETRY o RETURN.

```
CASE ERROR() = 108 .OR. ERROR() = 372          && El archivo está en uso en otra
                                                  estación
   merror = .F.
   mretry = .F.
   ACTIVATE WINDOW monitor
   @ 1, 3 SAY "La base de datos solicitada está en uso en otra etación de tra-
          bajo"
   @ 3, 3 SAY "Puede intentar de nuevo abrir la base de datos o salir"
   @ 4, 3 SAY "e intentar la ejecución del programa más tarde."
   @ 6, 3 SAY "¿Desea intentar abrir el archivo de nuevo? (Y/N)";
      GET mretry PICTURE "Y"
   READ
   DEACTIVATE WINDOW monitor
   IF mretry
      RETRY
   ELSE
      merror = .T.
      RETURN
   ENDIF
```

En este procedimiento, la variable de memoria MRETRY es una variable local que capta la respuesta del usuario a la pregunta y determina si el procedimiento de captación de errores regresa al programa invocador con RETRY o RETURN, esta variable no es utilizada por el programa que llama y no necesita inicializarse en él.

Además, puede añadir una orden DISPLAY USERS al procedimiento de captación de errores para visualizar una lista de los usuarios conectados actualmente a la red. Como último recurso, el usuario frustrado puede llamar a estas personas para averiguar lo que están haciendo y el tiempo que van a tardar en terminar. La orden

DISPLAY USERS lista el nombre de identificación de los usuarios establecidos en el sistema operativo de red, no el nombre de identificación de dBASE IV.

Un procedimiento de captación de errores que solicite al usuario lo que debe hacer si el archivo de base de datos no puede abrirse en el modo exclusivo funciona bien junto con los programas de mantenimiento de archivo que permiten a los usuarios seleccionar los archivos a procesar. Por ejemplo, podría volver a escribir el programa de reindexación y empaquetamiento de modo que permita que los usuarios elijan de antemano los archivos a procesar, como el programa NWBACK2.PRG descrito en el Capítulo 27, o de modo que se detenga cada archivo de base de datos y solicite si debe o no procesar el archivo.

La Figura 29-1 ilustra una versión del programa de reindexación que utiliza la estrategia anterior. Este programa supone que el archivo Archivos, que almacena la lista del nombre de las bases de datos y claves de índice, incluye un campo TAG que sirve para el mismo fin que el campo TAG del catálogo de dBASE IV estándar (le permite marcar un registro para identificarlo como el que debería procesarse posteriormente). El programa NWINDEX5.PRG examina el archivo Archivos y pide al usuario si procesa cada una de las bases de datos. El bucle anidado DO WHILE examina el grupo de los registros del archivo Archivos que representan a la base de datos actual; si el usuario quiere procesar este archivo, el programa marca cada registro en un grupo colocando la letra "P" en el campo TAG (una base de datos tiene un registro en el archivo Archivos para cada etiqueta de índices en el archivo de producción .MDX). Si se han marcado o no los registros, el puntero de registro del archivo Archivos se posiciona en la primera entrada de la siguiente base de datos cuando finaliza el bucle.

Después de solicitar la entrada del usuario para cada una de las bases de datos, el programa establece un filtro que selecciona solamente los registros marcados, basados en el contenido del campo TAG. El bucle DO WHILE examina el archivo Archivos y procesa cada una de las bases de datos seleccionadas (puesto que el programa puede saltar alguno de los registros marcados, el bucle DO WHILE es preferible al bucle SCAN, ya que le permite controlar el movimiento del puntero de registro a través del archivo de forma explícita). Pueden surgir dos posibles condiciones de error si la base de datos a procesar está abierta en otra estación de trabajo (la orden DELETE FILE que borra el archivo de producción .MDX puede fallar, produciendo el mensaje de error "No puede borrar un archivo abierto" ("Cannot erase open file") o la orden USE EXCLUSIVE puede fallar si otro usuario abre la base de datos después de ejecutar la orden DELETE FILE.

El programa de reindexación asume que estos errores son tratados por el programa de captación de errores NWERROR.PRG que establece MERROR a .T. si cualquier condición de error persiste después de un número de intentos determinado. En ambos casos, éste llama a un procedimiento listado llamado ERRMSG, que construye una lista de las bases de datos que no han sido procesadas, examina la siguiente base de datos del archivo Archivos, y le lleva al principio del bucle DO WHILE. Antes de que finalice el programa, la lista de archivos que ha sido omitida, almacenada en la variable de memoria MSKIPPED, es visualizada por una llamada al procedimiento MESSAGE4 descrito en el Capítulo 25.


```
* NWINDEX5.PRG
* PROGRAMA PARA EMPAQUETAR Y/O REINDEXAR TODAS LAS BASES DE DATOS
* ULTIMA REVISION: 08/06/89 M. LISKIN

DO Scrnhead WITH "", "Empaqueta y Reindexa todas las bases de datos"

DEFINE WINDOW utility FROM 5,10 TO 20,70
ACTIVATE WINDOW utility

mpack = .T.                          && ¿Empaquetar todas las bases de datos?
mindex = .T.                         && ¿Reindexar todas las bases de datos?
@ 1, 5 SAY "¿Desea eliminar los registros marcados? (Y/N)" GET mpack;
       PICTURE "Y"
@ 3, 5 SAY "¿Desea reindexar todas las bases de datos? (Y/N)" GET mindex;
       PICTURE "Y"
READ

IF mpack .OR. mindex

   SELECT Filelist                   && Area de trabajo del archivo Archivos
   REPLACE ALL Tag WITH " "          && Repone la señal de reindexación
   INDEX ON Database TAG Database
   GOTO TOP
   CLEAR
   @  1,10 SAY "Por favor elija los archivos a procesar:"
   mline = 3                         && Línea actual de la pantalla

   SCAN
      IF mline = 13                  && Si la ventana visualizada está llena,
         @ 3, 5 CLEAR                &&    borra el área de visualización
         mline = 3                   &&    repone la línea de la pantalla
         ENDIF
         mprocess = .T.              && ¿Procesa este archivo?
         @ mline,10 SAY "Procesar" + TRIM(Database) + "? (Y/N)"
         @ mline,50 GET mprocess PICTURE "Y"
         READ
         mline = mline + 1           && Incrementa la línea de la pantalla
         mdatabase = Database        && Nombre de la base de datos actual
         DO WHILE Database = mdatabase

         IF mprocess
            REPLACE Tag WITH "P"     && Marca la base de datos
         ENDIF
         SKIP
      ENDDO
   ENDSCAN
```

Figura 29-1.

Programa de reindexación y empaquetamiento multiusuario.

```
SET FILTER TO Tag = "P"          && Procesa los archivos seleccionados
GOTO TOP
mdatabase = " "                  && Ultima base de datos procesada
mskipped = ""                    && Archivos saltados

DO WHILE .NOT. EOF()
   merror = .F.                  && ¿Error de uso exclusivo?
   mtagname = TRIM(Tagname)      && Nombre de la etiqueta
   mindexkey = TRIM(Indexkey)    && Expresión clave de índice

IF Database <> mdatabase
   mdatabase   = TRIM(Database)  && Ultima base de datos procesada
   mindexfile = TRIM(Database) + ".MDX"     && Nombre del archivo MDX
   DELETE FILE &mindexfile       && Elimina el archivo MDX
   IF merror                     && Si el archivo MDX está abierto ya
      DO Errmsg
      LOCATE FOR Database <> mdatabase   && Busca la siguiente base de datos
      LOOP
   ENDIF
   USE (database) EXCLUSIVE      && Abre el archivo de base de datos
   IF merror                     && La base de datos ya está abierta
      DO Errmsg
      LOCATE FOR Database <> mdatabase   && Busca la siguiente base de datos
      LOOP
   ENDIF
   CLEAR
   DO Center WITH 6, 60, "", "Trabajando con " + DBF()
   IF mpack                      && Si el usuario elige empaquetar,
      SET TALK ON
      PACK                       &&    sólo empaqueta
      SET TALK OFF
      ENDIF
   ENDIF

   SELECT 2
   SET TALK ON
   INDEX ON &mindexkey .TAG & mtagname && Reconstruye la etiqueta de índices
   SET TALK OFF
   SELECT Filelist
   SKIP

  ENDDO

ENDIF.
```

Figura 29-1.

Programa de reindexación y empaquetamiento multiusuario *(continuación)*.


```
CLOSE DATABASES
DEACTIVATE WINDOW utility
RELEASE WINDOW utility

* Visualiza el mensaje de error si los archivos se han saltado
IF LEN(mskipped) > 0
   mskipped = LEFT(mskipped, LEN(mskipped)-2)   && Quita la última coma y el
                                                   espacio
   DO Message4 WITH "Los siguientes archivos se han saltado porque " +;
      "estaban en uso en otra estacion de trabajo: ", mskipped

ENDIF

RETURN

* Construye una lista de los archivos saltados para el mensaje de error
PROCEDURE Errmsg
mskipped = mskipped + mdatabase + ","
RETURN
* Fin del procedimiento Errmsg

* Fin del programa NWINDEX5.PRG
```

Figura 29-1.

Programa de reindexación y empaquetamiento multiusuario *(continuación)*.

Bloqueo de archivos compartidos

En una verdadera aplicación de base de datos multiusuario, los programas de consulta y actualización deben abrir archivos en modo compartido y permitir que más de un usuario actualice el mismo archivo simultáneamente. Cuando abre un archivo en modo compartido, es responsabilidad suya evitar los conflictos potenciales que puedan surgir al bloquear todo el archivo o registros individuales siempre que sea necesario. Para ayudarle a proteger los datos compartidos, dBASE IV bloquea automáticamente los archivos o registros afectados siempre que ejcuta una orden que requiere un bloqueo, y elimina éste después de finalizar la ejecución de la orden. Las órdenes y operaciones que requieren un bloqueo de archivo dependen en parte del parámetro LOCK. Cuando este parámetro está en ON, como es su valor por omisión, las órdenes listadas en la Tabla 29-1 (y las opciones de menú equivalentes) activan el bloqueo de archivo automáticamente. Cambiando el parámetro LOCK a OFF, puede desactivar el bloqueo automático de archivo para las órdenes que leen datos: órdenes que leen y escriben archivos, tales como INDEX, bloquean el archivo solamente durante la fase de escritura. Cuando dBASE IV está leyendo datos, desbloquea el archivo.

Si cambia el estado del parámetro LOCK a OFF, es responsabilidad suya asegurar que usuarios simultáneos no puedan realizar actualizaciones que dejen inconsistencia en los informes o en los resultados de los cálculos realizados sobre los datos.

Orden	Entidad bloqueada	Desactivada por SET LOCK OFF	Desactivada por SET LOCK OFF durante la fase de lectura
@... GET / READ	Registro	No	
APPEND FROM	Archivo	No	
APPEND BLANK	Registro	No	
AVERAGE	Archivo	No	
BROWSE	Registro	No	
CALCULATE	Archivo	Si	
CHANGE	Registro	No	
COPY TAG	Archivo		Si
COPY	Archivo		Si
COPY STRUCTURE	Archivo		Si
COUNT	Archivo	Si	
DELETE (un registro)	Registro	No	
DELETE	Archivo	No	
EDIT	Registro	No	
INDEX	Archivo		Si
JOIN	Archivo		Si
LABEL	Archivo	Si	
RECALL (one record)	Registro	No	
RECALL	Archivo	No	
REPLACE (one record)	Registro	No	
REPLACE	Archivo	No	
REPORT	Archivo	Si	
SET CATALOG ON	Archivo	No	
SORT	Archivo		Si
SUM	Archivo	Si	
TOTAL	Archivo		Si
UPDATE	Archivo	No	

Tabla 29-1.

Ordenes que activan un bloqueo de archivo automático.

Por ejemplo, si permite que el usuario de una estación de trabajo imprima un informe mientras otro ejecuta un programa que utiliza una orden REPLACE o @... GET para cambiar los campos del mismo archivo, debe ser consciente de que los datos impresos en cualquier lugar del informe pueden diferir, dependiendo de si los cambios han tenido lugar antes o después del informe impreso.

Desde el punto indicativo o desde dentro de un programa de dBASE IV, puede bloquear un archivo explícitamente abriéndolo y luego llamando a la función FLOCK. A diferencia de otras funciones de dBASE IV, excepto las correspondientes funciones de bloqueo de registro, la función FLOCK no sólo devuelve un valor (.T. si el archivo especificado puede ser bloqueado o .F. si ya estuviera bloqueado por otro usuario), sino también realiza una acción: intenta bloquear el archivo. Desde el punto

indicativo, la forma más fácil de llamar a la función FLOCK es utilizar la orden ? para visualizar el resultado de la evaluación, del siguiente modo:

```
? = FLOCK()
```

En un programa, puede preceder esta orden con SET CONSOLE OFF de modo que no visualice información extraña en la pantalla; pero es más conveniente evaluar la función FLOCK almacenando su valor en una variable de memoria temporal "flotante", del siguiente modo:

```
mx FLOCK("Customer")
```

Puede bloquear un archivo en un área de trabajo no seleccionada proporcionando la referencia al área de trabajo (el número del área de trabajo, la letra, o el alias del archivo) como entrada opcional.

Si la llamada a la función FLOCK no logra bloquear la base de datos especificada debido a que el archivo o al menos un registro del archivo está bloqueado en otra estación de trabajo, la función FLOCK toma el valor .F. y el archivo no es bloqueado. No hay forma de determinar la parte del archivo que no está disponible: podría estar ocupado varias horas mientras los índices requeridos para los informes se construyen o un programa de dBASE IV lleva a cabo una actualización. No obstante, el archivo puede estar disponible en unos cuantos segundos, si el usuario está utilizando un programa de consulta y actualización para hacer cambios rápidos. Un usuario que trabaja a nivel de mandato puede que decida esperar e intentarlo de nuevo en unos minutos o localizar quién está utilizando el archivo y preguntarle a esa persona lo que está haciendo y el tiempo que tardará en terminar.

Para hacer frente a esta variedad de posibilidades, el programa que solicita un bloqueo de archivo no debería evaluar la función FLOCK una vez y darlo por imposible si el archivo no puede ser bloqueado; pero no debe intentarlo de forma indefinida. En la mayoría de los casos, una estrategia análoga a la descrita en el apartado anterior es más apropiada: el programa debería evaluar la función FLOCK varias veces durante algún período de tiempo arbitrario y luego preguntar al operador si continúa intentándolo o informarle de que lo intente más tarde. No obstante, se requiere una secuencia de órdenes algo diferente. Cuando una llamada a la función FLOCK no logra bloquear un archivo, dBASE IV no considera este error (la función devuelve el valor .F., que resulta ser perfectamente válido). En consecuencia, no puede hacer uso del parámetro REPROCESS para especificar un número arbitrario de intentos antes de llamar a la rutina de captación de errores. En su lugar, debe llamar a un procedimiento o función definida por el usuario que dé por resultado el valor .T. para el programa invocador si éste logra bloquear el archivo, o .F. en caso contrario, tal como la siguiente:

```
FUNCTION Filelock

mlocked = .T.
mcount = 1
```

```
DO WHILE mco...    < 100 .AND. .NOT. FLOCK()
   mcount = mcount + 1
ENDDO
IF .NOT. FLOCK()
   mlocked = .F.
   DO Message3 WITH "La base de datos solicitada no está disponible. " +;
      "Por favor inténtelo de nuevo más tarde."
ENDIF

RETURN mlocked
```

El bucle DO WHILE de este programa vuelve a evaluar la función FLOCK cien veces (experimentando un poco con su propio sistema bajo una red típica, descubrirá el límite superior más apropiado para MCOUNT). Si el archivo no puede bloquearse, la función visualiza un mensaje de error para explicar la situación al usuario y configurar MLOCKED, la variable que la función devuelve al programa invocador, a .F. Observe que también puede determinar si el archivo ha sido bloqueado con éxito comprobando el valor de MCOUNT después de la sentencia ENDDO. Si el bucle DO WHILE finaliza debido a que dBASE IV no bloquea la base de datos después de intentarlo 100 veces, MCOUNT tendrá el valor 101; si una de las llamadas anteriores a FLOCK bloqueó el archivo, MCOUNT tendrá un valor comprendido entre 1 y 100. No obstante, la comprobación de FLOCK da por resultado una estructura IF más legible que claramente indica la condición que se está verificando (intenta por última vez bloquear el archivo). Si el retardo causado por las repetidas llamadas a FLOCK es considerable y en consecuencia percibido por los usuarios, puede visualizar un mensaje en la parte inferior de la pantalla o en una ventana de aparición súbita para describir lo que está haciendo el programa. El siguiente procedimiento incorpora esta mejora:

```
PROCEDURE Filelock

ACTIVATE WINDOW monitor

mlocked = .T.
mcount = 1
DO Center WITH 1, 60, msmessages, "Requiriendo acceso a " + DBF()
DO Center WITH 2, 60, msmessages, "Pulse cualquier tecla para interrumpir la
operación"
DO WHILE mcount <= 100 .AND. .NOT. FLOCK() .AND. INKEY() = 0
   mcount = mcount + 1
ENDDO
IF .NOT. FLOCK()
   mlocked = .F.

   DO Center WITH 4, 60, msmessages,;
      "La base de datos solicitada no está disponible."
   DO Center WITH 5, 60, msmessages,;
      "Por favor inténtelo de nuevo más tarde."
   WAIT
ENDIF

DEACTIVATE WINDOW monitor

RETURN
```


También puede dar al usuario la opción de continuar intentando el bloqueo del archivo.

Si lleva a cabo la secuencia de bloqueo con una función definida por el usuario, puede llamar a esta función en una estructura IF que trate la posible condición de error del siguiente modo:

```
IF .NOT. Filelock()
   * Ordenes para tratar la condición de error y salir
ENDIF
* Ordenes para operar sobre el archivo bloqueado

o

IF Filelock()
   * Ordenes para operar sobre el archivo bloqueado
ELSE
   * Ordenes para tratar la condición de error y salir
ENDIF
```

Este uso es idéntico al análogo llamado por la función FLOCK incorporada. No obstante, si su rutina de captación de errores incluye órdenes que no están permitidas en las funciones definidas por el usuario, debe utilizar un procedimiento en su lugar. En el sistema National Widgets, el procedimiento CENTER usado para visualizar los mensajes de estado y el mensaje de error utiliza una macro para expresar los códigos de color pasados al procedimiento a través del parámetro MCOLOR. Debido a que las macros están prohibidas en las funciones definidas por el usuario (y en cualquier procedimiento o programa llamado por una función definida por el usuario), la rutina de bloqueo de archivos se estructura como un procedimiento.

Si es imposible bloquear una base de datos, el programa que llama, no el procedimiento o función de bloqueo, debe decidir qué hacer a continuación. En algunos casos, particularmente, en un programa que permita que los usuarios seleccionen los archivos a procesar, es posible abrir y bloquear una base de datos diferente y continuar. No obstante, incluso si no hay otra alternativa más que salir al programa de menú que lo llamó, el programa debe cerrar primero cualquier base de datos abierta, desactivar los menús y las ventanas, y eliminar las variables de memoria públicas, menús, y definiciones de ventana de la memoria. La función o procedimiento que lleva a cabo la secuencia de bloqueo de archivo debe, por tanto, devolver al programa que llama una variable de memoria que registre el resultado del intento, en lugar de actuar sobre este resultado.

Cuando bloquea un archivo de forma explícita con una llamada a la función FLOCK, también debe acordarse de desbloquearla cuando finalice la operación. dBA-

SE IV desbloquea automáticamente una base de datos cuando la cierra, o puede desbloquear el archivo (o cualquier registro bloqueado del archivo) en el área de trabajo seleccionada dejándola abierta utilizando una orden UNLOCK:

```
UNLOCK
```

Advierta que aunque debe llamar a una *función* para bloquear a un archivo, el archivo se desbloquea con una *orden*. Para eliminar todos los archivos (y registros) bloqueados de su estación de trabajo, en todas las áreas de trabajo, utilice:

```
UNLOCK ALL
```

Bloqueo de registros compartidos

Cuando utilice órdenes o programas de dBASE IV que operen sobre toda la base de datos, tal como un programa para reindexar y empaquetar o un programa de utilidad que reconstruya el balance de los clientes basándose en detalles de las transacciones, la necesidad de proteger la integridad de los datos abriendo un archivo en modo exclusivo o bloqueando el archivo vale más que el inconveniente de que los usuarios tengan que esperar para acceder al archivo. Estas operaciones globales, llevadas a cabo por procedimientos de actualización secuenciales o de mantenimiento de bases de datos, tales como los programas descritos en el Capítulo 26, se realizan de forma infrecuente comparadas con las rutinas de actualización diarias llevadas a cabo por los programas que permiten que los usuarios introduzcan, consulten, o editen registros simultáneamente. Mientras que dos usuarios no tengan permiso para editar los mismos registros de forma simultánea, es posible abrir con seguridad una o más bases de datos para uso compartido y permitir que más de una estación de trabajo ejecute un programa de consulta y actualización.

Cuando actualiza una base de datos en la pantalla Hojear o Editar, o a través de un programa a medida, dBASE IV bloquea el registro actual automáticamente cuando escribe una modificación en cualquier campo. La visualización de datos con órdenes @ ... GET, o incluso moviendo el cursor a través de los campos, no activa el bloqueo automático de registros. Si hay abierta más de una base de datos, dBASE IV también bloquea cualquier registro en el área de trabajo no seleccionada que esté enlazada con el registro actual con SET RELATION. Para aprovecharse de este bloqueo automático en el programa de consulta y actualización de clientes, tendría que enlazar el archivo Clientes con el archivo Histórico de Transacciones con SET RELATION y SET SKIP, en lugar de usar una orden SEEK explícita para encontrar los registros que coinciden (no es necesario bloquear los registros del archivo Histórico de Pedidos, que no pueden actualizarse mediante este programa). dBASE IV elimina el bloqueo automático, desbloqueando todos los registros bloqueados en ambas bases de datos, cuando el puntero de registro se mueve por el archivo principal (en este caso, el archivo Clientes).

Si un usuario intenta hacer algún cambio en un registro que está bloqueado en otra estación de trabajo, o en cualquier registro de un archivo que está bloqueado en otra estación de trabajo, dBASE IV inicia una estrategia de bloqueo similar a su estrategia de bloqueo de archivo. Si el parámetro REPROCESS mantiene su valor por omisión, cero, el intento de bloquear un registro que ya esté en uso en otra estación de trabajo produce el error "El registro está en uso" ("Record in use by another"). Cuando el parámetro REPROCESS tiene un valor distinto de cero, dBASE IV vuelve a intentar el bloqueo el número de veces especificado antes de generar la condición de error. En este caso, puede utilizar la rutina de captación de errores similar a la que trataba los errores que resultaban de intentos sin éxitos de abrir un archivo para uso exclusivo. Un ejemplo típico es el siguiente:

```
CASE ERROR() = 109 .OR. ERROR() = 373        && El registro está en uso en
                                                otra estación
   merror = .F.
   mretry = .F.
   ACTIVATE WINDOW monitor
   @ 1, 3 SAY "Este registro está siendo actualizado en otra estación de tra-
              bajo"
   @ 3, 3 SAY "Puede intentar de nuevo el acceso al registro o puede intentar-
              lo"
   @ 4, 3 SAY "de nuevo más tarde."
   @ 6, 3 SAY "¿Desea intentar el acceso al registro de nuevo? (Y/N)";
          GET mretry PICTURE "Y"
   READ
   IF mretry
      RETRY
   ELSE
      merror = .T.
      RETURN
   ENDIF
```

A menudo tiene sentido permitir que el usuario decida intentarlo de nuevo, incluso si no permite la elección equivalente en la rutina que trata el error "El archivo está en uso" ("File in use by another"). Muchas operaciones que requieren un bloqueo de archivo consumen mucho tiempo, por razones que el usuario no puede comprender, mientras que un bloqueo de registro es probable que se realice en poco tiempo, cuando el usuario que ha bloqueado el registro en otra estación de trabajo termine la edición del mismo. Si la red es rápida y los usuarios comprenden la importancia de almacenar un registro tan pronto como se hayan completado los cambios, raras veces estará esperando mucho tiempo antes de que esté disponible cualquier registro.

Cuando dos usuarios visualizan el mismo registro en la pantalla Hojear o Editar, cada uno está viendo una copia independiente de los datos, que están almacenados en la memoria, en su estación de trabajo. dBASE IV, por omisión, visualiza la actualización realizada en otras estaciones de trabajo que reflejan los cambios realizados solamente cuando los usuarios mueven el puntero de registro y desbloquean el registro actualizado. Puede utilizar la orden SET REFRESH para configurar dBASE IV para refrescar la visualización en un intervalo especificado, independientemente del

tiempo que permanezca el registro bloqueado. Sin embargo, para hacer esto, debe utilizar la orden CONVERT para añadir un campo especial del estado de la red a la estructura del archivo de cada base de datos. dBASE puede refrescar la pantalla solamente cuando actualiza una base de datos en la pantalla Editar u Hojear (con o sin un archivo de formato que represente la entrada de datos), no cuando capta los datos con un programa a medida con una serie de órdenes @... GET seguidas por una READ.

En la mayoría de las aplicaciones a medida, debe abandonar las posibilidades de bloqueo de registro automáticas de dBASE IV y bloquear los registros de forma explícita antes de permitir que el usuario edite los datos. Cuando actualiza un archivo que tenga más de un campo, los usuarios pueden perder algún tiempo estudiando el contenido del registro antes de empezar a escribir los cambios, y en desplazar el cursor a través de todos los campos que preceden al que debe modificar. Bloqueando el registro de antemano, puede asegurar su disponibilidad en el momento en que el usuario empiece la edición. Algunas opciones, tal como **Marcar** en el programa de consulta y actualización de clientes, pueden visualizar mensajes de error o hacer algunas preguntas antes de permitir que el usuario realice la eliminación. Por las mismas razones definidas en el Capítulo 28, no con la función RLOCK, también denominada LOCK (los ejemplos de este libro utilizan RLOCK para diferenciarla de FLOCK, que bloquea un archivo entero) puede bloquear registros individuales. Al igual que la función de bloqueo de archivo correspondiente, RLOCK intenta bloquear el registro especificado o los registros y devuelve el valor: .T. si el registro puede bloquearse o .F. si el registro o el archivo completo está ya bloqueado por otro usuario. Utilizando la orden ? para visualizar el resultado de la evaluación, puede llamar a la función RLOCK o almacenar el valor en una variable de memoria temporal "flotante" así:

```
mx = RLOCK()
```

También es posible bloquear más de un registro a la vez listando el número de registro, como una cadena de caracteres encerrada entre comillas, como argumento de la función RLOCK, junto con el alias del archivo. Por ejemplo, la siguiente orden bloquearía los registros 1, 2 y 3 del archivo Clientes:

```
mx = RLOCK("1, 2, 3", "Customer")
```

Además puede especificar el alias para bloquear los registros en un área de trabajo no seleccionada. La orden anterior bloquearía los registros 1, 2 y 3 del archivo de Clientes si éste estuviese abierto en cualquier área de trabajo (y tiene asignado el alias "CUSTOMER").

dBASE IV siempre bloquea los registros que están enlazados a un registro bloqueado con SET RELATION y SET SKIP. Si ha enlazado los archivos Histórico de Transacciones y Clientes con SET RELATION y SET SKIP, el bloqueo del registro del cliente actual bloquearía automáticamente las transacciones correspondientes. Sin embargo, como se dijo en el Capítulo 24, esta estrategia es relativamente ineficiente. Si un usuario no necesita editar transacciones en una sesión de trabajo dada, o si el nivel de acceso del usuario no permite que el usuario edite transacciones, los enlaces

retardan el movimiento del puntero de registro a través del archivo Clientes. En cambio, puede bloquear todos los registros de transacciones relacionados de forma explícita antes de llamar al programa de actualización de transacciones. Para añadir el bloqueo de registro al programa de consulta y actualización de clientes, podría modificar la sección CASE del procedimiento CUSTUPD que llama al programa de actualización de transacción, como sigue:

```
CASE moption = "TRANSACT"
   SELECT Tranhist
   SEEK Customer -> Account
   IF .NOT. FOUND()
      DO Message3 WITH "Este cliente no tiene transacciones"
   ELSE
      SCAN WHILE Account = Customer -> Account
           = RLOCK()
      ENDSCAN
      DO NWtxnupd
   ENDIF
   SELECT Customer
```

El programa también debe tratar el error que ocurriría si fue imposible bloquear uno o más registros de transacciones debido a que otro usuario estaba actualizando el archivo Histórico de Transacciones. Una técnica para hacer esto se pone de manifiesto en los programas de actualización descritos más tarde en este capítulo.

Puesto que dBASE IV le permite bloquear más de un registro a la vez, éste no desbloquea un registro cuando bloquea otro. Con la misma orden UNLOCK utilizada para eliminar el bloqueo del archivo, puede desbloquear un único registro de forma explícita, y desbloquear todos los registros presentes en el área de trabajo actual con:

```
UNLOCK ALL
```

Para desbloquear los registros en un área de trabajo no seleccionada, debe especificar en la orden UNLOCK el área de trabajo:

```
UNLOCK ALL IN Tranhist
```

Al igual que no es necesario bloquear un archivo para ver o imprimir su contenido, sólo debe bloquear un registro cuando esté siendo actualizado. El bloqueo de registros individuales proporciona a los usuarios la máxima flexibilidad utilizando la aplicación de forma concurrente, ya que las operaciones que no alteran ningún dato pueden proceder como si no existieran otros usuarios. La mayoría de actualizaciones pueden llevarse a cabo sin conflicto, ya que es raro que dos usuarios intenten acceder al mismo registro del mismo archivo simultáneamente.

UN PROGRAMA DE CONSULTA Y ACTUALIZACION MULTIUSUARIO

Un programa de consulta y actualización diseñado para su uso en un entorno de red debe verificar que ninguna de las bases de datos necesarias esté abierta para uso exclusivo en otra estación de trabajo. En la mayoría de los casos, no necesita bloquear todo el archivo, sino que debe bloquear los registros siempre que el programa permita que los usuarios los actualicen (editen, eliminen o recuperen). Bloqueando el registro actual de forma explícita cuando el usuario selecciona la opción de menú correspondiente, puede garantizar que el registro esté disponible cuando llegue la hora de realizar la actualización. En el programa de consulta y actualización de clientes, las opciones **Buscar, Siguiente, Anterior, Primero, Ultimo, Hojear, Imprimir** y **Pedidos**, que solamente visualizan datos, no requieren un bloqueo de registro. Las opciones **Cambiar, Introducir, Marcar** y **Desmarcar** requieren que bloquee el registro actual del archivo Clientes, y la opción **Transacciones** requiere que bloquee el registro de cliente actual y el registro de transacciones correspondiente también.

La Figura 29-2 lista NWCUPD5.PRG, una nueva versión del programa de consulta y actualización de clientes que incorpora las medidas de seguridad descritas en el Capítulo 28, así como nuevas posibilidades multiusuario. Este programa supone que el parámetro EXCLUSIVE está en OFF, puede añadir una orden SET EXCLUSIVE OFF al programa principal de la aplicación para asegurar que todas las bases de datos puedan ser abiertas en modo compartido, a menos que un programa, tal como el programa de reindexación y empaquetamiento, solicite de forma explícita el acceso exclusivo. Este también supone que la orden ON ERROR anterior establece como programa de tratamiento de errores de la aplicación a NWERROR.PRG descrito en el Capítulo 25. La orden SET REPROCESS del principio del programa hace que dBASE IV intente las operaciones de red diez veces antes de presentar el mensaje de error. Podría colocar esta orden en el programa de arranque principal si no necesita cambiar el número de intentos dentro de la aplicación.

```
* NWCUPD5.PRG
* PROGRAMA DE CONSULTA Y ACTUALIZACION MULTIARCHIVO CON SEGURIDAD
* ULTIMA REVISION: 07/31/89 M. LISKIN

SET DELETED OFF                 && Muestra los registros marcados
SET REPROCESS TO 10             && Hace 10 intentos después de un error de red

* Abre las bases de datos
merror = .F.                    && El bloqueo del archivo o registro no se
                                   consigue
USE NWcust ORDER Account ALIAS Customer
USE NWtxn IN 2 ORDER Account ALIAS Transact
```

Figura 29-2.

Un programa de consulta y actualización multiusuario.


```
USE NWthist IN 3 ORDER Account ALIAS Tranhist
USE NWohist IN 4 ORDER Account ALIAS Ordhist
USE NWinvent IN 5 ORDER Partnumber ALIAS Inventory
USE NWcodes IN 6 ORDER Codetype ALIAS Codes
SELECT Ordhist
SET RELATION TO Partnumber INTO Inventory
SELECT Customer

IF merror                       && Si el archivo no está disponible
   CLOSE DATABASES              &&    porque ya está abierto
   SET DELETED ON               &&    para uso exclusivo
   SET REPROCESS TO 0
   RETURN                       && Regresa al programa invocador
ENDIF

DECLARE mrecord[18]             && Campos de Clientes

DO Menudef                      && Define las ventanas y los menús

CLEAR
GOTO TOP                        && Regresa al primer cliente
DO Custsays                     && Visualiza el texto de fondo
DO Custdata                     && Visualiza el primer cliente

ACTIVATE MENU Custupd           && Activa el menú de opciones

CLOSE DATABASES
SET DELETED ON                  && Oculta los registros marcados
SET REPROCESS TO 0
RELEASE MENU Custupd
RETURN

*************************************************************************

* Define las ventanas y visualiza los menús
PROCEDURE Menudef

* Define las ventanas para recoger los campos memo
DEFINE WINDOW comment1 FROM 11,8 TO 15,76

DEFINE WINDOW comment2 FROM 11,8 TO 20,76

* Define la ventana para responder a las preguntas
DEFINE WINDOW question FROM 10,10 TO 14,70 DOUBLE

* Define la ventana para visualizar el registro de pedidos
DEFINE WINDOW ordlist FROM 3,0 TO 21,79
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.

```
* Define la ventana para actualizar las transacciones
DEFINE WINDOW transact FROM 3,0 TO 24,79 NONE

* Define la línea de menús principal para la consulta y actualización del
cliente
DEFINE MENU custupd
DEFINE PAD Search OF custupd AT 22,1 PROMPT "Buscar";
   MESSAGE "Busca un cliente por el código de cuenta"
DEFINE PAD Next OF custupd AT 22,10 PROMPT "Siguiente";
   MESSAGE "Visualiza el siguiente cliente"
DEFINE PAD First OF custupd AT 22,21 PROMPT "Primero";
   MESSAGE "Visualiza el primer cliente"
DEFINE PAD Change OF custupd AT 22,29 PROMPT "Cambiar";
   MESSAGE "Cambia los datos del cliente visualizado"
DEFINE PAD Delete OF custupd AT 22,38 PROMPT "Marcar";
   MESSAGE "Marca el cliente visualizado"
DEFINE PAD Orders OF custupd AT 22,48 PROMPT "Pedidos";
   MESSAGE "Consulta de pedidos"
DEFINE PAD Print OF custupd AT 22,63 PROMPT "Imprimir";
   MESSAGE "Imprime información del cliente visualizado"
DEFINE PAD Browse OF custupd AT 23,1 PROMPT "Hojear";
   MESSAGE "Hojear listado de códigos de cuenta, nombre de compañías o del
           contacto"
DEFINE PAD Previous OF custupd AT 23,10 PROMPT "Anterior";
   MESSAGE "Visualiza el cliente anterior"
DEFINE PAD Last OF custupd AT 23,21 PROMPT "Ultimo";
   MESSAGE "Visualiza el último cliente"
DEFINE PAD Enter OF custupd AT 23,29 PROMPT "Introducir";
   MESSAGE "Introducir un nuevo cliente"
DEFINE PAD Recover OF custupd AT 23,38 PROMPT "Desmarcar";
   MESSAGE "Recupera (desmarca) el cliente visualizado"
DEFINE PAD Transact OF custupd AT 23,48 PROMPT "Transacciones";
   MESSAGE "Consulta y actualización de transacciones"
DEFINE PAD Memo OF custupd AT 23,63 PROMPT "Memo";
   MESSAGE "Introducir o editar texto en campos memo"
DEFINE PAD Quit OF custupd AT 23,70 PROMPT "Salir";
   MESSAGE "Regresar al menú principal"

ON SELECTION PAD Search OF custupd DO Custupd
ON SELECTION PAD Next OF custupd DO Custupd
ON SELECTION PAD Previous OF custupd DO Custupd
ON SELECTION PAD First OF custupd DO Custupd
ON SELECTION PAD Last OF custupd DO Custupd
ON SELECTION PAD Enter OF custupd DO Acctest WITH "Custupd", 4
ON SELECTION PAD Print OF custupd DO Custupd
ON SELECTION PAD Change OF custupd DO Acctest WITH "Custupd", 4
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.


```
ON SELECTION PAD Delete OF custupd DO Acctest WITH "Custupd", 1
ON SELECTION PAD Recover OF custupd DO Custupd
ON SELECTION PAD Browse OF custupd DO Custupd
ON SELECTION PAD Transact OF custupd DO Acctest WITH "Custupd", 2
ON SELECTION PAD Orders OF custupd DO Acctest WITH "Custupd", 2
ON SELECTION PAD Memo OF custupd DO Acctest WITH "Custupd", 4
ON SELECTION PAD Quit OF custupd DEACTIVATE MENU

RETURN
* Fin del procedimiento Menudef

* Ejecuta las opciones del menú
PROCEDURE Custupd

moption = PAD()

IF moption $ "CHANGE/ENTER/DELETE/RECOVER/TRANSACT/MEMO"
   mlocked = .T.
   DO Reclock
   IF .NOT. mlocked
      RETURN
   ENDIF
ENDIF

DO CASE

   CASE moption = "SEARCH"
      maccount = SPACE(10)

      ACTIVATE WINDOW Question
      @ 1,2 SAY "Introduzca el código de cuenta del cliente" GET maccount
FUNCTION "A!";
            VALID SEEK(TRIM(maccount));
            ERROR TRIM(maccount) + "no se encuentra en el archivo Clientes"
      READ
      DEACTIVATE WINDOW Question

  CASE moption = "NEXT"
     IF .NOT. EOF()
        SKIP
     ENDIF
     IF EOF()
        DO Message3 WITH "Este es el último cliente"
        GOTO BOTTOM
     ENDIF
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.

```
CASE moption = "PREVIOUS"
   IF .NOT. BOF()
      SKIP -1
   ENDIF
   IF BOF()
      DO Message3 WITH "Este es el primer cliente"
      GOTO TOP
   ENDIF

CASE moption = "FIRST"
   GOTO TOP

CASE moption = "LAST"
   GOTO BOTTOM

CASE moption = "ENTER"
   maccount = SPACE(10)
   ACTIVATE WINDOW question
   @ 1,2 SAY "Introduzca un código de cuenta único para el nuevo cliente"
         GET maccount PICTURE "@A!";
         VALID maccount = " " .OR. Isunique(maccount);
         ERROR TRIM(maccount) + "ya existe en el archivo Clientes"
   READ

   DEACTIVATE WINDOW question
   IF maccount = " "
      RETURN
   ELSE
      GOTO RECCOUNT()          && Ultimo registro introducido
      mrecord[1] = maccount    && Código de cuenta introducido antes
      mrecord[5] = City        && Repite la última entrada
      mrecord[6] = msstate     && Estado por omisión
      mrecord[8] = msareacode  && Código del área por omisión
      STORE DATE() TO mcrecord[12], mcrecord[13]
      * Inicia los restantes elementos del array a blancos
      mrecord[2] = SPACE(30)
      STORE SPACE(25) TO mcrecord[3], mcrecord[4], mcrecord[10]
      mcrecord[5] = SPACE(20)
      mcrecord[6] = " "
      mcrecord[7] = SPACE(10)
      mcrecord[9] = SPACE(8)
      mcrecord[11] = SPACE(100)
      mcrecord[13] = .F.
      STORE 0.00 TO mcrecord[15], mcrecord[16], mcrecord[17],;
         mmcrecord[18]
   ENDIF
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.


```
CASE moption = "CHANGE"
   COPY NEXT 1 TO ARRAY mcrecord

CASE moption = "DELETE"
   IF Totinv - Totpmt <> 0
      DO Message3 WITH "Balance distinto de 0.00"
      RETURN
   ENDIF
   IF FOUND("Transact")
      DO Message3 WITH "Existen transacciones ... ales"
      RETURN
   ENDIF
   mconfirm = .F.
   DO Yesno WITH mconfirm,;
      "¿Está seguro de que desea marcar este cliente?"
   IF mconfirm

      DELETE
      @ 0,10 SAY "* Marcado *"
      DO Message3 WITH "Cliente marcado"
   ENDIF
   UNLOCK
   RETURN

CASE moption = "RECOVER"
   RECALL
   @ 0,10 SAY SPACE(11)
   DO Message3 WITH "Cliente desmarcado"
   UNLOCK
   RETURN

CASE moption = "BROWSE"
   BROWSE FIELDS Account, Company, Contact, City, State;
      NOMENU NOAPPEND NOEDIT NODELETE WIDTH 20 WINDOW browscrn

CASE moption = "PRINT"
   DO Custprnt

CASE moption = "TRANSACT"
   SELECT Tranhist
   SEEK Customer->Account
   IF .NOT. FOUND()
      DO Message3 WITH "Este cliente no tiene transacciones"
   ELSE
      SCAN WHILE Account = Customer->Account
         mlocked = .T.
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.

```
            DO Reclock
            IF .NOT. mlocked
               UNLOCK ALL
               SELECT Customer
               RETURN
            ENDIF
         ENDSCAN
         DO NWtxnupd
      ENDIF
      SELECT Customer

   CASE moption = "ORDERS"
      SELECT Ordhist
      SEEK Customer->Account
      IF .NOT. FOUND()
         DO Message3 WITH "Este cliente no ha realizado pedidos"
      ELSE
         * Copia los campos requeridos en un archivo temporal
         SET FIELDS TO Account, Customer->Account, Category, Partnumber,;
            Inventory->Descrip, Quantity, Price, Subtotal, Discount,;
            Invamount
         COPY TO NWtemp WHILE Account = Customer->Account
         SET FIELDS TO
         * Visualiza los datos a partir de un archivo temporal
         SELECT 10
         USE NWtemp
         BROWSE WINDOW ordlist COMPRESS NOMENU NOAPPEND NODELETE WIDTH 15;
            FIELDS Category, Partnumber, Descrip, Quantity, Price,;
            Subtotal, Discount, Invamount
         USE
      ENDIF
      SELECT Customer
      RETURN

   CASE moption = "MEMO"
      IF ACCESS() <= 4
         @ 11, 8 GET Comments OPEN WINDOW comment1
      ELSE
         @ 11, 8 GET Comments OPEN WINDOW comment2
      ENDIF
      READ

ENDCASE

IF moption = "BROWSE"
   DO Custsays
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.


```
ENDIF
DO Custdata
IF moption $ "CHANGE/ENTER"
   DO Arrgets

   READ
   IF moption = "ENTER"
      APPEND BLANK
      REPLACE Account WITH mcrecord[1]
   ENDIF
   REPLACE Company WITH mcrecord[2], Address1 WITH mcrecord[3],;
      Address2 WITH mcrecord[4], City WITH mcrecord[5],;
      State WITH mcrecord[6], Zip WITH mcrecord[7],;
      Areacode WITH mcrecord[8], Telephone WITH mcrecord[9],;
      Contact WITH mcrecord[10], Equipment WITH mcrecord[11]
   IF ACCESS <= 2
      REPLACE Firstorder WITH mcrecord[12],;
         Lastorder WITH mcrecord[13], Creditok WITH mcrecord[14],;
         Ytdinv WITH mcrecord[15], Ytdpmt WITH mcrecord[16],;
         Totinv WITH mcrecord[17], Totpmt WITH mcrecord[18]
   ENDIF
   * Vuelve a calcular y a visualizar el balance
   @ 20,67 SAY Totinv-Totpmt PICTURE "@ $ 999,999.99"
ELSE
   DO Custgets
   CLEAR GETS
ENDIF

UNLOCK ALL

RETURN
* Fin del procedimiento Custupd

* Visualiza el texto de los campos
PROCEDURE Custsays

* Ordenes para visualizar el texto de los campos

* Fin del procedimiento Custsays
* Visualiza los campos
PROCEDURE Custdata

@ 0,10 SAY IIF(DELETED(), "* Marcado *", SPACE(11))
@ 2,13 SAY Account COLOR ,&msfields.
@ 4,13 SAY Company COLOR ,&msfields.
@ 5,13 SAY Address1 COLOR ,&msfields.
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.

```
@ 6,13 SAY Address2 COLOR ,&msfields.
@ 7,13 SAY City COLOR ,&msfields.
@ 7,52 SAY State COLOR ,&msfields.
@ 7,62 SAY Zip COLOR ,&msfields.
@ 4,59 SAY Telephone COLOR ,&msfields.
@ 9,13 SAY Equipment COLOR ,&msfields.
IF ACCESS() <= 4
   @ 11, 8 SAY Comments WINDOW comment1
ELSE
   @ 11, 8 SAY Comments WINDOW comment2
ENDIF
IF ACCESS() <= 4
   @ 17,15 SAY Firstorder COLOR ,&msfields.
   @ 18,15 SAY Lastorder COLOR ,&msfields.
   @ 17,68 SAY Totinv PICTURE "99,999.99" COLOR, &msfields.
   @ 17,39 SAY Ytdinv PICTURE "99,999.99" COLOR, &msfields.
   @ 18,39 SAY Ytdpmt PICTURE "99,999.99" COLOR, &msfields.
   @ 18,68 SAY Totpmt PICTURE "99,999.99" COLOR, &msfields.
   @ 20,15 SAY Creditok PICTURE "Y" COLOR, &msfields.
   @ 20,67 SAY Totinv - Totpmt PICTURE "@ $ 999,999.99"
ENDIF
* Vuelve a visualizar el código del área y el estado
DO Descrip

RETURN
* Fin del procedimiento Custdata

* Capta los datos
PROCEDURE Arrgets

@ 0,10 SAY IIF(DELETED(), "* Marcado *", SPACE(11))
@ 2,13 GET mcrecord[1] COLOR, &msfields.
CLEAR GETS
@ 4,13 GET mcrecord[2] COLOR, &msfields.
@ 5,13 GET mcrecord[3] VALID mcrecord[3] <> " " ;
       ERROR "La primera línea de dirección no debe estar en blanco";
       COLOR, &msfields.
@ 6,13 GET mcrecord[4] COLOR, &msfields.
@ 7,13 GET mcrecord[5] VALID mcrecord[5] <> " " ;
       ERROR "La ciudad no debe estar en blanco";
       COLOR, &msfields.
@ 7,52 GET mcrecord[6] PICTURE "@A!";
       VALID SEEK("S" + mcrecord[6], "Codes");
       ERROR "El estado debe ser una abreviatura válida de 2 letras";
       COLOR ,&msfields.
@ 7,62 GET mcrecord[7] PICTURE "@A!" VALID Zipok(mcrecord[7], " ");
       WHEN mcrecord[6] <> " "
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.


```
@ 4,53 GET mcrecord[8] PICTURE "999"
       VALID SEEK("A" + mcrecord[8], "Codes");
       ERROR "No es un código de área válido";
       COLOR &msfields.
@ 4,59 GET mcrecord[9] PICTURE "999-9999" COLOR ,&msfields.
@ 6,52 GET mcrecord[10];
       VALID mcrecord[2] <> " " .OR. mcrecord[10] <> " ";
       ERROR "El nombre de la compañía y el contacto no pueden estar en blanco a la vez"
       COLOR &msfields.
@ 9,13 GET mcrecord[11] PICTURE "@S62" COLOR, &sfields.
IF ACCESS() <= 2
   @ 17,15 GET mrecord[12] WHEN mcrecord[12] = { / / };
           COLOR &msfields.
   @ 18,15 GET mcrecord[13] VALID mcrecord[13] >= mcrecord[12];
           .AND. mcrecord[13] <= DATE() ;
           ERROR "El último pedido no puede ser anterior al primero o posterior a hoy";
           MESSAGE "Use la fecha de la factura, no la del cheque de los pedidos pago anticipado";
           COLOR &msfields.
   @ 17,39 GET mcrecord[15] PICTURE "99,999.99" COLOR &msfields.
   @ 17,68 GET mcrecord[17] PICTURE "99,999.99" COLOR &msfields.
   @ 18,39 GET mcrecord[16] PICTURE "99,999.99" COLOR &msfields.
   @ 18,68 GET mcrecord[18] PICTURE "99,999.99" COLOR &msfields.
   @ 20,15 GET mcrecord[15] PICTURE "Y" COLOR &msfields.
ENDIF

RETURN
* Fin del procedimiento Arrgets

* Visualiza las descripciones del archivo de códigos
PROCEDURE Descrip

SELECT Codes
SEEK "A" + Customer->Areacode    && Consulta el código del área
@ 5,44 SAY Descrip               && Código del área
SEEK "S" + Customer->State       && Consulta la abreviatura del estado
@ 8,44 SAY Descrip               && Nombre del estado
SELECT Customer

RETURN
* Fin del procedimiento Descrip

* Imprime el cliente actual
PROCEDURE Custprnt
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación)*.

```
* Ordenes para imprimir el cliente actual

RETURN
* Fin del procedimiento Custprnt

* Fin del programa NWCUPD5.PRG
```

Figura 29-2.

Un programa de consulta y actualización multiusuario *(continuación).*

El programa confía en la rutina de tratamiento de errores para detectar y responder a las condiciones de error iniciadas al intentar abrir una base de datos que ya está abierta para uso exclusivo en otra estación de trabajo. Antes de abrir cualquier archivo, inicializa la variable de memoria, MERROR, que es utilizada por la rutina de tratamiento de errores para informar de estos errores al programa que llama. Después de las órdenes USE que abren los archivos requeridos, el programa vuelve a comprobar el valor de MERROR; si es .T. (debido a que estuviera abierta alguna base de datos), éste cierra los archivos abiertos y regresa al menú.

Para determinar si el archivo Clientes o Histórico de Transacciones está bloqueado en otra estación de trabajo, puede añadir una condición al principio del programa. Si otro usuario ha bloqueado cualquier archivo, dBASE IV permite que el programa de actualización de clientes abra el archivo, pero no bloquea ningún registro individual. Esta versión del programa de actualización no incorpora esa condición y permite que el usuarsio vea e imprima los registros, incluso si es posible llevar a cabo la actualización. Si desea bloquear estos archivos de antemano, puede hacer lo siguiente:

```
IF ACCESS() <= 2
   SELECT Tranhist
   DO Filelock
   IF merror
      CLOSE DATABASE
      SET DELETED ON
      SET REPROCESS TO 0
      RETURN
   ENDIF
   SELECT Customer
ENDIF

IF ACCESS() <= 4
   DO Filelock
   IF merror
      CLOSE DATABASE
      SET DELETED ON
      SET REPROCESS TO 0
      RETURN
   ENDIF
ENDIF

UNLOCK
```


Para el archivo Histórico de Transacciones y luego para el archivo Clientes, esta secuencia de órdenes comprueba el nivel de acceso del usuario, y si el usuario actual está autorizado para actualizar la base de datos, llama al procedimiento FILELOCK para intentar bloquear el archivo. Si el archivo no puede ser bloqueado, el programa cierra todas las bases de datos abiertas, restaura los parámetros DELETED y REPROCESS a sus valores por omisión, y sale al programa invocador. En el supuesto de que cualquier intento bloquee de hecho una base de datos, la orden UNLOCK asegura que ningún archivo permanecerá bloqueado. Una tercera alternativa es permitir que los usuarios decidan si proceden o no con el programa si uno de los archivos no puede ser bloqueado. Por ejemplo, si el usuario ha pensado sólo en ver o imprimir los registros, no sería necesario bloquear ningún archivo, y si el usuario no necesita editar las transacciones, sólo es necesario bloquear el archivo Clientes.

El programa de actualización de clientes intenta bloquear todos los registros llamando al procedimiento RECLOCK, que funciona del mismo modo que el procedimiento FILELOCK descrito anteriormente en este capítulo: realiza el número de intentos apropiado y luego permite que el usuario decida qué hacer a continuación. El procedimiento RECLOCK, que está localizado en el archivo de procedimientos principal de la aplicación, es el siguiente:

```
PROCEDURE Reclock

ACTIVATE WINDOW monitor

mlocked = .T.
DO WHILE .T.
   mcount = 1
   DO Center WITH 1, 60, msmessages, "Demandando acceso -- " +;
      "Pulse cualquier tecla para interrumpir la operación"
   DO WHILE mcount <= 100 .AND. .NOT. RLOCK() .AND. INKEY() = 0
      mcount = mcount + 1
   ENDDO
   IF RLOCK()
      EXIT
   ELSE
      CLEAR
      mretry = .F.
      @ 1, 3 SAY "Este registro está siendo actualizado en otra estación de
                  trabajo"
      @ 3, 3 SAY "Puede intentar de nuevo el acceso al registro o puede inten-
                  tarlo "
      @ 4, 3 SAY "de nuevo más tarde."
      @ 6, 3 SAY "¿Desea intentar el acceso al registro de nuevo? (Y/N)";
             GET mretry PICTURE "Y"
      READ
```

```
      IF .NOT. mretry
        mlocked = .F.
        EXIT
      ENDIF
    ENDIF
ENDDO

DEACTIVATE WINDOW monitor

RETURN
```

Para permitir que el usuario decida si vuelve a intentar o no el bloqueo de un registro que no ha tenido éxito, este procedimiento lleva a cabo una condición en un bucle "indefinido" controlado por la sentencia DO WHILE .T. Después de cada intento, sale del bucle, permitiendo que MLOCKED permanezca en su valor inicial .T. si el registro ha sido bloqueado con éxito; en cualquier otro caso, pide al usuario si lo intenta de nuevo. Si el usuario responde "No", pone la variable MLOCKED a .F. y sale del bucle DO WHILE. Nótese que aunque MLOCKED tiene asignado ya un valor en el procedimiento RECLOCK, debe estar inicializada en el programa que llama (en este caso, al principio del procedimiento CUSTUPD), para hacerla disponible después de que finalice el procedimiento RECLOCK.

El programa de actualización multiusuario inicializa MLOCKED y llama a RECLOCK en el procedimiento CUSTUPD, antes de la estructura DO CASE que procesa las opciones individuales, si el usuario selecciona la opción **Cambiar, Introducir, Eliminar, Recuperar** o **Transacciones**, o la nueva opción **Memo.** Si el procedimiento RECLOCK establece MLOCKED a .F., que indica que el registro actual no podría ser bloqueado, una orden RETURN sale del procedimiento CUSTUPD inmediatamente y permite que el usuario haga otra selección a partir del menú de opciones. Observe la interacción entre los tests de nivel de acceso del usuasrio y las órdenes de bloqueo de registro en esta versión del programa de actualización. El test de nivel de acceso se lleva a cabo primero, cuando el usuario selecciona una opción de menú. Si el usuario puede ejecutar la opción seleccionada, el programa continúa para determinar si el registro actual está disponible para la edición.

Las secciones CASE del procedimiento CUSTUPD que tratan las opciones **Eliminar** y **Recuperar** sólo requieren algunas pequeñas modificaciones: la adición de una orden UNLOCK para eliminar el bloqueo de registro después de que el programa ejecute las órdenes DELETE o RECALL que actualizan el registro actual. La opción **Transacciones** utiliza un bucle SCAN, como el descrito anteriormente en este capítulo, para bloquear todos los registros de transacciones que coinciden con el registro del cliente actual. Si no es así, el programa ejecuta una orden UNLOCK ALL para desbloquear cualquier registro que estuviera bloqueado con éxito y sale al procedimiento CUSTUPD. En su lugar, puede elegir bloquear los registros de transacciones de forma individual en el programa NWTXNUPD.PRG, antes de que se actualicen.

Esta versión del programa de actualización no permite que el usuario edite un registro de clientes directamente. Las opciones **Introducir** y **Cambiar** captan los datos en un array llamado MCRECORD (de "registro de cliente"), que ha sido declarado al principio del programa, y luego transfiere los cambios al archivo Clientes con una serie de órdenes REPLACE. Las órdenes CASE que tratan la opción **Cambiar** utilizan COPY TO ARRAY para transferir el contenido de cada campo (excepto el del campo memo COMMENTS) del registro actual al elemento del array correspondiente. Cuando utiliza un array para captar datos, no puede editar los campos memo, ya que no puede crear una variable de memoria o un elemento del array del tipo memo. El programa de actualización multiusuario incluye una opción del menú aparte, **Memo,** que permite que los usuarios editen los campos memo (sujeto a las restricciones usuales según el nivel de acceso).

La orden CASE que trata la opción **Introducir** debe asignar el valor inicial apropiado a los elementos del array. El primer elemento, que será el código de cuenta, supone que el valor se ha introducido en MACCOUNT, y los elementos que corresponden a los campos STATE y AREACODE (MCRECORD[6] y MCRECORD[8]) se asignan a los valores por omisión de la aplicación estándar almacenados en las variables de opciones globales MSSTATE y MSAREACODE. Para repetir el contenido del campo CITY de la entrada anterior, el programa posiciona el archivo Clientes en el último registro introducido con GOTO RECCOUNT() (no puede utilizar GOTO BOTTOM cuando un índice está activo, ya que el último registro en el orden indexado no es necesariamente el que se encuentra en el final físico del archivo) y utiliza el contenido del campo CITY de este registro como el valor inicial del elemento del array correspondiente, MCRECORD[5]. Las dos fechas (MCRECORD[12] y MCRECORD[13]) se inicializan con la fecha actual, expresada con la función DATE, y a los restantes elementos se les asigna el valor cero o espacios blancos.

Esta versión del programa de actualización utiliza tres procedimientos independientes para visualizar los datos y mensajes del cliente. Como es usual, el procedimiento CUSTSAYS visualiza los mensajes con órdenes @... SAY. Puesto que el usuario nunca edita los campos de cliente directamente (excepto el campo memo), los datos también se visualizan con órdenes @... SAY, por un procedimiento denominado CUSTDATA. Este procedimiento es llamado al principio del programa para visualizar el primer registro de cliente y después mover el puntero de registro o regresar al programa de actualización de transacciones. Un nuevo procedimiento, ARRGETS, que es análogo en su función al procedimiento CUSTGETS utilizado en las versiones anteriores del programa de actualización, capta los elementos del array MCRECORD. Los procedimientos CUSTSAYS y CUSTDATA visualizan los campos financieros y sus mensajes solamente si el nivel de acceso del usuario (un valor 4 o inferior) da permiso para consultar estos datos, y el procedimiento ARRGETS capta estos datos solamente si la función ACCESS da por resultado 1 ó 2, el nivel más alto requerido para modificar estos campos.

Después de la estructura DO CASE que procesa la selección del usuario, el programa llama al procedimiento apropiado para visualizar o editar los datos. Si el usuario introduce un nuevo registro de clientes, añade un registro en blanco al archivo Clientes y cumplimenta el campo ACCOUNT según el valor del primer elemento del array. No tiene que bloquear el nuevo registro ya que la orden APPEND BLANK activa el bloqueo de registro automáticamente, que en este caso es suficiente para proteger la nueva entrada. Para cualquier cambio o nuevo registro, el programa reemplaza todos los campos con los elementos del array correspondiente (puesto que esta

versión del programa no permite que el usuario edite el código de cuenta del cliente de un registro existente, no es necesario reemplazar este campo si el usuario selecciona la opción **Cambiar**). Este paso es realizado por dos órdenes REPLACE independientes: la segunda, que transfiere los datos en los campos financieros, se ejecuta solamente si el usuario tiene permiso para editar estos campos. Si el usuario actual sólo tiene prioridad R/O sobre estos campos, la orden REPLACE generaría el error "Nivel de acceso no autorizado" ("Unauthorized access level").

El último paso del procedimiento CUSTUPD es una orden UNLOCK ALL, que elimina todos los bloqueos de registro (en los archivos Transacciones y Clientes) antes de que finalice el procedimiento. Si lo desea, puede modificar el programa NWTXNUPD.PRG con líneas similares para recoger los campos de transacciones en los elementos de un array y luego transferir los cambios al registro del archivo Histórico de Transacciones actual.

PROCESO DE TRANSACCIONES

dBASE IV proporciona una facilidad adicional, *el proceso de transacciones*, para ayudarle a proteger las bases de datos cuando un procedimiento de actualización de archivo complejo no logra completarse normalmente. El término "transacción", utilizado en este sentido, denota cualquier secuencia de órdenes que actualiza una o más bases de datos. Las órdenes que constituyen la transacción se identifican encerrándolas entre las órdenes BEGIN TRANSACTION y END TRANSACTION.

Cuando comienza una transacción, dBASE IV almacena el estado actual del entorno de trabajo (las bases de datos y los índices abiertos) en un *archivo de registro de operaciones* en el disco. En un sistema monousuario, el archivo de registro de operaciones se denomina TRANSLOG.LOG; en una red, dBASE IV crea un archivo de registro diferente para cada usuario y le asigna el nombre de red del usuario (el nombre utilizado para conectarse con la red, no el nombre de identificación introducido para acceder a dBASE IV si está en efecto un sistema de seguridad) como primer nombre del archivo de registro de operaciones. Cuando las órdenes actualizan los registros en la transacción, dBASE IV graba en el archivo de registro de operaciones el contenido de cada uno de los registros afectados, antes y después de cualquier cambio, de modo que el archivo puede ser restaurado a su estado original si la transacción no se completa normalmente. Después de ejecutar la sentencia END TRANSACTION, dBASE IV borra el archivo de registro de operaciones; debe llevar a cabo cualquier test que sea necesario para verificar que la transacción se ha completado correctamnete antes de finalizar.

En una transacción están permitidas la mayoría de las órdenes de dBASE IV. Las órdenes prohibidas incluyen aquellas que cierran archivos o escriben encima de los archivos de disco existentes, ya que estas órdenes no permiten que dBASE IV invierta las actualizaciones realizadas por la transacción CLEAR ALL, CLOSE ALL, CLOSE DATABASES, CLOSE INDEX, DELETE FILE, ERASE, INSERT, MODIFY STRUCTURE, PACK, RENAME, y ZAP siempre están prohibidas en una transacción, y COPY, COPY STRUCTURE, CREATE, EXPORT, IMPORT, INDEX, JOIN, SET CATALOG, SET INDEX, SORT, TOTAL y USE también generan errores si cierran archivos o escriben encima de los archivos existentes. Puesto que no puede cerrar un archivo, debe seleccionar una nueva área de trabajo para cada archivo que abra durante una transacción, incluso si no enlaza los archivos o no los utiliza juntos de ninguna forma.

Cuando está en progreso una transacción, dBASE IV se ejecuta de forma más lenta debido al tiempo necesario para grabar todas las actualizaciones en el archivo de registro de operaciones. Observe que aunque no existe límite en el número de órdenes que puede ejecutar entre las sentencias BEGIN TRANSACTION y END TRANSACTION, una transacción larga daría por resultado un archivo de registro de operaciones muy extenso, ya que este archivo almacena el estado de cada uno de los registros antes y después de que sea alterado en la transacción. Por ejemplo, si añade un proceso de transacciones a una utilidad de mantenimiento de base de datos que calcula cada uno de los balances del cliente según el contenido del archivo Histórico de Transacciones puede identificar las órdenes que actualizan cada registro de clientes como una transacción.

Para restaurar los archivos a su estado original basándose en el archivo de registro de operaciones, debe utilizar la orden ROLLBACK. Mientras una transacción está en progreso (antes de la orden END TRANSACTION), la orden ROLLBACK restaura todas las bases de datos abiertas. Después de la orden END TRANSACTION, puede especificar un nombre de base de datos en la orden ROLLBACK para restaurar una sola base de datos. En general, debería echar atrás los archivos de forma individual solamente si la orden ROLLBACK no logra restaurar todos los archivos y no dispone de una copia de seguridad reciente de todas las bases de datos.

Comprobando el valor de la función COMPLETED, que toma el valor .T. si la transacción se ha completado con éxito, o .F. en caso contrario, puede determinar si la transacción se ha completado normalmente, después de la sentencia END TRANSACTION. Basándose en este resultado, puede decidir si ejecuta o no una orden ROLLBACK. Además, puede determinar si esta orden se ha ejecutado con éxito comprobando el valor de la función ROLLBACK, que devuelve el valor lógico .T. si la base de datos ha sido restaurada con éxito, o .F. en caso contrario. Si la función ROLLBACK da por resultado .F. después de una orden ROLLBACK, su programa debería informar a los usuarios que la orden ROLLBACK se ha ejecutado sin éxito y, por tanto, deben consultar al administrador de la red antes de continuar utilizando la aplicación. Una secuencia típica de órdenes puede ser la siguiente:

```
BEGIN TRANSACTION
* Commands to update files
END TRANSACTION

IF .NOT. COMPLETED()
   ACTIVATE WINDOW monitor
   DO Center WITH 1, 60, msmessages,;
      "Attempting to restore files -- Please do not interrumpt"
   ROLLBACK
   IF ROLLBACK()
      DO Center WITH "Archivos restaurados" + mspresskey
      DEACTIVATE WINDOW monitor
```

```
   ELSE
      CLEAR
      DO Center WITH 1, 60, mswarning, "*** PRECAUCION ***"
      DO Center WITH 3, 60, msmessages,;
         "Los archivos no pueden restaurarse"
      DO Center WITH 4, 60, msmessages,;
         "Por favor consulte inmediatamente con el administrador de la red"
      @ 5, 0
      WAIT
      CLOSE DATABASES
      QUIT
   ENDIF
ENDIF
```

Este uso del proceso de transacciones está pensado para que sea posible invertir una actualización del archivo completada parcialmente. Por ejemplo, en un programa que pasa datos de un grupo de registros del archivo de Transacciones al archivo de Clientes y luego pone a cero el campo POSTED en los registros del archivo de Transacciones si no está disponible una transacción a mitad de camino (quizá porque está bloqueada en otra estación de trabajo), puede que desee "deshacer el paso" del grupo entero e intentar la operación de nuevo más tarde. Una alternativa, que se ilustra en el programa de actualización de clientes, es bloquear todos los registros de transacciones de antemano para garantizar que todos serán accesibles. Pero si el grupo es extenso, este proceso puede ser un poco lento.

Si el sistema falla a mitad de una transacción, dBASE IV visualiza una ventana de diálogo de aparición súbita cuando el usuario arranca el programa otra vez para avisarle que está pendiente todavía una transacción: "** PRECAUCION ** Se ha encontrado una transacción incompleta". La ventana de diálogo tiene una opción, **Continuar**, que al seleccionarla dBASE IV intenta automáticamente deshacer la transacción. Puesto que la orden ROLLBACK puede que no sea capaz de restaurar las bases de datos si se han dañado físicamente en la cadena del sistema, el proceso de transacción no debería utilizarse como sustituto de las copias de seguridad.

Después de producirse un fallo, puede restaurar las bases de datos a partir de las copias de seguridad o intentar una orden ROLLBACK selectiva de tantas bases de datos como sea posible desde la estación de trabajo del usuario afectada. Si la orden ROLLBACK no puede recuperar la base de datos, puede ser más fácil corregir las inconsistencias editando los registros desde el punto indicativo, o determinar el daño físico con una utilidad tal como las Norton Utilities, que pueden recuperar el archivo desde la última copia de seguridad y volver a introducir todos los cambios y los nuevos registros. Después de hacer eso, debe utilizar la orden RESET para extraer las marcas internas que se establecen cuando se actualiza el archivo en una transacción para que los usuarios trabajen con el archivo. Antes de ejecutar la orden RESET, debe abrir el archivo en modo exclusivo.

El proceso de transacción está pensado principalmente para su uso en los programas de dBASE IV diseñados para ser ejecutados en grandes redes, cuando los archivos de bases de datos son susceptibles a los daños causados por fallos en cualquier parte de la red así como en la estación de trabajo que ejecuta el programa, y donde el número de actualizaciones realizadas cada día es lo suficientemente grande para que la restauración de las bases de datos desde las copias de seguridad de los días anteriores antes de la caída del sistema o del error cometido por el usuario sea impracticable. No obstante, también puede incluir un proceso de transacciones en aplicaciones monousuarios para proteger los archivos de algún tipo de inconsistencia interna que pueda ocurrir si un proceso largo que actualiza un archivo basado en los datos de otro no se completa normalmente.

COORDINACION Y COOPERACION DEL USUARIO

La operación con éxito de una aplicación en un entorno multiusuario requiere un alto nivel de cooperación entre los usuarios del sistema y los programas de dBASE IV. Sus programas deberían tratar algunos de los conflictos potenciales que pueden presentarse y deberían evitar que los usuarios realicen operaciones concurrentes que tengan peligro de dañar las bases de datos y deberían visualizar mensajes de error informativos siempre que el acceso a un programa u orden sea denegado por cualquier razón. A pesar de eso, el grado de cooperación y comunicación entre los usuarios determinará lo satisfachos (o frustrados) que están con el comportamiento y flexibilidad de la aplicación.

Este concepto no es nuevo o único completamente para los sistemas de red. En una oficina que sólo disponga de una computadora, los usuarios puede que compitan por el tiempo de máquina y debatan sobre qué proceso de datos es más urgente en un momento dado. Algunos de estos problemas se solucionan con una red, ya que varios operadores podrán acceder no sólo a la computadora sino también a los mismos archivos de datos. Por esta razón, puede que sea una sorpresa para aquellos usuarios que no están familiarizados con la programación en red no poder acceder precisamente a los archivos que desean en un momento dado: aquellos archivos pueden haber sido abiertos para uso exclusivo en otra estación de trabajo.

Además, todos los usuarios deberían ser conscientes de las operaciones que están restringidas o producen inconvenientes para otros usuarios de la red. En un entorno LAN, es importante ejecutar los procesos que consumen mucho tiempo y cierran los archivos enteros en momentos en que otros usuarios no están esperando utilizar los mismos archivos. Los usuarios deben darse cuenta también de que si más de una persona está realizando un proceso intenso en disco, tal como la indexación de una base de datos o la lectura de un archivo para imprimir un informe, esto hará que la red se comporte con lentitud, incluso para estaciones de trabajo que no estén utilizando dBASE IV. En consecuencia, puede que sea necesaria alguna planificación de las operaciones que se ejecutarán cuando se presenten estos inconvenientes a otros usuarios.

Todo esto requiere una actitud de cooperación y un conocimiento profundo de las consideraciones de la programación de la red que muchas personas, aun cuando estén familiarizadas con el uso de las microcomputadoras en general, puede que no posean. Es esencial que el diseñador o el administrador del sistema comunique esta información a sus usuarios, en términos no técnicos que puedan entender con facilidad, tanto personalmente como en la documentación impresa que acompaña a la aplicación.

APENDICE B

Ordenes

ORDENES POR TIPOS

Diseño de archivos y formatos.

CREATE	CREATE/MODIFY SCREEN
CREATE/MODIFY APPLICATION	CREATE VIEW FROM ENVIRONMENT
CREATE/MODIFY LABEL	MODIFY COMMAND
CREATE/MODIFY QUERY	MODIFY STYLE
CREATE/MODIFY REPORT	MODIFY STRUCTURE

Data Base Maintenance and Reorganization

APPEND BLANK	APPEND MEMO
APPEND FROM ARRAY	CLOSE
COPY MEMO	PACK
COPY STRUCTURE	RECALL
COPY STRUCTURE EXTENDED	REPLACE
COPY TO	SELECT
COPY TO ARRAY	TOTAL
CREATE FROM	UPDATE
DELETE	USE
INSERT BLANK	ZAP
JOIN	

Record Sequencing

COPY INDEXES	INDEX
COPY TAG	REINDEX
DELETE TAG	SORT

Memory, Memory Variables, Arrays and Macros

APPEND FROM ARRAY	PRIVATE
CLEAR ALL	PUBLIC
CLEAR FIELDS	RELEASE
CLEAR MEMORY	RESTORE
CLEAR TYPEAHEAD	RESTORE MACROS
DECLARE	SAVE MACROS
DISPLAY/LIST MEMORY	SAVE
PLAY MACRO	STORWE/=

Mode Selection

ASSIST	QUIT
HELP	SET

Networking and Security

CONVERT	RESET
DISPLAY/LIST USERS	ROLLBACK
LOGOUT	UNLOCK
PROTECT	

Programming, Program Structure, and Flow Control

BEGIN TRANSACTION... END TRANSACTION	PLAY MACRO
CANCEL	PRINTJOB...ENDPRINTJOB
COMPILE	PROCEDURE
DEGUB	RESET
DO	RESUME
DO CASE...ENDCASE	RETRY
DO WHILE...ENDDO	RETURN
EXIT	ROLLBACK
FUNCTION	SCAN...ENDSCAN
IF...ELSE...ENDIF	SUSPEND
LOOP	TEXT...ENDTEXT
NOTE/*/&&	WAIT
PARAMETERS	

Accessing External Programa

CALL	RELEASE MODULE
LOAD	RUN/!

Record Pointer Positioning and Search

CONTINUE	LOCATE
FIND	SEEK
GO/GOTO	SKIP

Import and Export

APPEND FROM	EXPORT
COPY INDEXES	IMPORT
COPY TO	

Disk File Management

COPY FILE	ERASE
DELETE FILE	RENAME
DIRECTORY/DIR	TYPE
DISPLAY FILES	

Data Collection

@... GET	EDIT
ACCEPT	INPUT
APPEND	INSERT
BROWSE	READ
CHANGE	

Data Display and Printing

?/??/???	EJECT
@... CLEAR	LABEL FORM
@... FILL	LIST
@... SAY	REPORT FORM
@... TO	SHOW MENU
CLEAR GETS	SHOW MENU
DISPLAY	SHOW POPUP
DISPLAY	

Calculation

AVERAGE	SUM
CLACULATE	TOTAL
COUNT	UPDATE
REPLACE	

Screen, Menu, and Window Control

ACTIVATE MENU
ACTIVATE POPUP
ACTIVATE SCREEN
ACTIVATE WINDOW
CLEAR
CLEAR MENUS
CLEAR POPUPS
CLEAR WINDOWS
DEACTIVATE MENU
DEACTIVATE POPUP
DEACTIVATE WINDOW
DEFINE BAR
DEFINE BOX
DEFINE MENU
DEFINE PAD
DEFINE POPUP
DEFINE WINDOW
DISPLAY/LIST MEMORY
MOVE WINDOW
ON PAD
ON SELECTION PAD
ON SELECTION POPUP
RESTORE WINDOW
SAVE WINDOW

Event Detection and Processing

EJECT PAGE
ON ERROR
ON ESCAPE
ON KEY
ON PAGE
ON READERROR

Environment Status

LIST/DISPLAY FILES
LIST/DISPLAY HISTORY
LIST/DISPLAY MEMORY
LIST/DISPLAY STATUS
LIST/DISPLAY STRUCTURE

SET COMMANDS

Environment

SET BELL
SET CENTURY
SET ESCAPE
SET EXACT
SET FULLPATH
SET FUNCTION
SET HELP
SET HISTORY
SET INSTRUCT
SET CONFIRM
SET DESIGN
SET NEAR
SET ODOMETER
SET SAFETY
SET SCOREBOARD
SET TALK
SET TYPEAHEAD

Data Base

SET AUTOSAVE
SET BLOCKSIZE
SET CARRY
SET DELETED
SET FIELDS
SET FILTER
SET FORMAT
SET INDEX
SET ORDER
SET RELATION
SET SKIP
SET UNIQUE
SET VIEW
SET WINDOW OF MEMO

Files

SET ALTERNATE
SET CATALOG
SET DEFAULT
SET PATH
SET TITLE

Data Display

SET CURRENCY
SET DATE
SET DECIMALS
SET DELIMITERS
SET HEADING
SET MARK
SET MEMOWIDTH
SET POINT
SET PRECISION
SET SEPARATOR

Output Device and Screen Control

SET BORDER
SET CLOCK
SET COLOR
SET CONSOLE
SET DEVICE
SET DISPLAY
SET HOURS
SET INTENSITY
SET MARGIN
SET MESSAGE
SET PRINTER
SET SPACE
SET STATUS

Programming and Debugging

SET DEBUG
SET DEVELOPMENT
SET ECHO
SET PROCEDURE
SET STEP
SET TRAP

Networking and Security

SET ENCRYPTION
SET EXCLUSIVE
SET LOCK
SET REFRESH
SET REPROCESS

SQL

SET PAUSE SET NEART SQL

SINTAXIS DE LAS ORDENES

?/?? [<*exp 1*> [PICTURE <*imagen*>] [FUNCTION <*funciones*>]
[AT <*columna*>][STYLE <*código de estilo*>] [, <*exp 2*> ...]...

??? [<*expC*>]

@ <*fila*>, <*columna*> [SAY <*exp*> [PICTURE <*imagen*>]
[FUNCTION <*funciones*>]]
[GET <*variable*> [PICTURE <*imagen*>] [FUNCTION <*funciones*>]
[RANGE <*exp 1*>, <*exp 2*>] [VALID <*condición*> [ERROR <*expC*>]]
WHEN <*condición*>] [DEFAULT <*exp*>][MESSAGE <*expC*>]]
[[OPEN] WINDOW <*nombre de la ventana*>]] COLOR
<*primer plano/fondo*>[, <*primer plano/fondo*>]]

@ <*fila 1*>, <*columna 1*> CLEAR [TO <*fila 2*>, <*columna 2*>

@ <*fila 1*>, <*columna 1*> CLEAR [TO <*fila 2*>, <*columna 2*>>
[COLOR <*primer plano/fondo*>]

@ <*fila 1*>, <*columna 1*> TO <*fila 2*>, <*columna 2*>
[DOUBLE/PANEL/ <*caracteres del contorno*>] [COLOR <*primer plano/
fondo*>]

ACCEPT [<*expC*> TO <*varmemo*>

ACTIVATE MENU <*nombre de menú*> [PAD <*nombre de opción*>]

ACTIVATE POPUP <*nombre popup*>

ACTIVATE SCREEN

ACTIVATE WINDOW <*lista nombre ventana*>/ALL

APPEND

APPEND BLANK

APPEND FROM ARRAY <*nombre del array*> [FOR <*condición*>]

APPEND FROM <*nombre de archivo*> [TYPE DBASEII/DIF/FW2/RPD/SDF/SYLK
WKS/DELIMITED [WITH <*delimitador*>/BLANK]] [FOR <*condición*>]

APPEND MEMO <*campo memo*> FROM >*nombre de archivo*> [OVERWRITE]

ASSIT

ABERAGE [<*lista expN*>][TO <*lista varmemo*> / ARRAY <*nombre de array*>]
[<*alcance*>] [FOR <*condición*> [WHILE <*condición*>]

BEGIN TRANSACTION [<*nombre de opción*>]...END TRANSACTION

BROWSE FIELDS [<*campo 1*>] [/R] [/ <*anchura de columna*>]
/ <*nombre1 del campo calculado 1*> = <*exp 1*> [, <*campo 2*>...]...]
[NOINIT] [NOFOLLOW] [NOAPPEND] [NOMENU] [NOEDIT] [NODELETE]
[NOCLEAR] [COMPRESS] [FORMAT] [LOCK <*columnas*>]
[WIDTH <*expN*>] [FREEZE <*campo*>] [WINDOW <*nombre ventana*>]

CALCULATE [<*lista exp*>] [TO <*lista de varmemo*> / ARRAY <*nombre de array*>]
[<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

Funciones permitidas en *lista exp:* AVG (<*expN*>), CNT(), MAX(<*exp*>), MIN (<*exp*>), NPV (<*interés*>, <*flujo de caja*>, <*inversión inicial*>), STD (<*expN*>), SUM (<*expN*>), VAR (<*expN*>)

CALL >*nombre del módulo*> [WHIT <*lista exp*>]

CANCEL

CHANGE [FIELDS <*lista de campos*>] [<*número de registro*>] [<*alcance*>]
[FOR <*condición*>] [WHILE <*condición*>] [NOINIT] [INOFOLLOW]
NOAPPEND] [NOMENU] [NODELETE] [NOCLEAR]

CLEAR ALL/FIELDS/GETS/MEMORY/MENUS/POPUPS/
TYPEAHEAD/WINDOWS

CLOSE ALL/ALTERNATE/DATABASES/FORMAT/INDEXES/
PROCEDURE

COMPILE <*nombre de archivo*> [RUNTIME]

CONVERT [TO <*expN*>]

COPY TO <*nombre archivo*> [TYPE DBASEII/DBMEMO3/DIF/EW2/RPD/SDF/
SYLK/WKS/DELIMITED [WITH <*delimitador*>/BLANK]]
FIELDS <*lista de campos*>[<*alcance*>] [FOR <*condición*>]
[WHILE <*condición*>]

COPY FILE <*archivo 1*> TO <*archivo 2*>

COPY INDEXES <*lista de archivos .NDX*> [TO <*nombre de archivo .MDX*>]

COPY MEMO <*campo memo*> TO <*nombre del archivo*> [ADDITIVE]

COPY STRUCTURE TO <*nombre de archivo*> [FIELDS <*lista de campos*>]

COPY TAG <*nombre de etiqueta*> [OF <*archivo .MDX*>] TO <*archivo .NDX*>

COPY TO <*nombre de archivo*> STRUCTURE EXTENDED

COPY TO ARRAY <*nombre de arrays*> [FIELDS <*lista de campos*>] [<*alcance*>]
[FOR <*condición*>] [WHILE <*condición*>]

COUNT [TO <*varmemo*>] [<*alcance*>] [FOR <*condición*>]
[WHILE <*condición*>]

CREATE <*nombre de base de datos*>

CREATE <*nombre de base de datos*> FROM <*archivo de estructura extendida*>

CREATE/MODIFY APPLICATION <*nombre de archivo de aplicación*>

CREATE/MODIFY LABEL <*nombre de formato de etiqueta*>

CREATE/MODIFY QUERY/VIEW <*nombre de archivo de consulta*>

CREATE/MODIFY REPORT <*nombre de formulario de informe*>

CREATE/MODIFY SCREEN <*nombre de formato de pantalla*>

CREATE VIEW <*nombre de archivo de visión*> FROM ENVIRONMENT

DEACTIVATE MENU

DEACTIVATE POPUP

DEACTIVATE WINDOW <*lista de nombres de ventana*>/ALL

DEBUG <*nombre de programa*>/<*nombre de procedimiento*> [WITH <*lista de parámetros*>]

DECLARE <*nombre del array 1*> [<*filas*> [, <*columnas*>]]
[, <*nombre del array 2*> [<*filas*> [, <*columnas*>]]]...

DEFINE BAR <*número de línea*> OF <*nombre popup*> PROMPT <*expC*>
[MESSAGE <*expC*>] [SKIP [FOR <*condición*>]]

DEFINE BOX FROM <*columna 1*> TO <*columna 2*> HEIGHT <*expN*>
[AT LINE <*línea*>] [SINGLE/DOUBLE/ <*caracteres del contorno*>]

DEFINE MENU <*nombre de menú*> [MESSAGE <*expC*>]

DEFINE PAD <*nombre de opción*> OF <*nombre de menú*> PROMPT <*expC*>
[AT <*fila*>, <*columna*>] [MESSAGE <*expC*>]

DEFINE POPUP <*nombre popup*> FROM <*fila 1*>, <*columna 1*>
[TO <*fila 2*>, <*columna 2*>] [PROMPT FIELD <*nombre de campo*>/ PROMPT
FILES [LIKE <*esquema*>]/PROMPT STRUCTURE] [MESSAGE <*expC*>]

DEFINE WINDOW <*nombre de la ventana*> FROM <*fila 1*>, <*columna 1*> TO
<fila 2>, <*columna 2*> [DOUBLE/PANEL/NONE/ <*caracteres del contorno*>
[COLOR [<*primer plano estándar/fondo estándar*>] [, <*primer plano realzado/fondo realzado*>] [, <*marco*>]]

DELETE [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

DELETE FILE <*nombre de archivo*>

DELETE TAG <*nombre de etiqueta 1*> [OF <*archivo índice .MDX*]/ <*archivo índice .NDX*> [, <*nombre de etiqueta 2*> ...]

DIRECTORY/DIR [[ON] <*unidad*> :] [[LIKE] [<*camino*>\] [<*esquema*>]

DISPLAY [<*lista exp*>] [FOR <*condición*>] [WHILE <*condición*>] [<*alcance*>]
[OFF] [TO PRINTER/TO FILE <*nombre de archivo*>]

DISPLAY FILES [LIKE <*esquema*>] [TO PRINTER/TO FILE <*nombre de archivo*>]

DISPLAY HISTORY [LAST <*expN*>] [TO PRINTER/TO FILE <*nombre de archivo*>]

DISPLAY MEMORY [TO PRINTER/TO FILE <*nombre de archivo*>]

DISPLAY STATUS [TO PRINTER/TO FILE <*nombre de archivo*>]

DISPLAY STRUCTURE [IN <*alias*>] [TO PRINTER/TO FILE <*nombre de archivo*>]

DISPLAY USERS

DO <*nombre de programa*>/<*nombre de procedimiento*> [WITH <*lista de parámetros*>]

DO CASE...ENDCASE

El formato general de la estructura es:

```
DO CASE
    CASE <condición 1>
        <sentencias del programa>...
    [CASE <condición 2>
        <sentencias del programa>...]
    [CASE <condición 3>
        <sentencias del programa>
    [<demás casos>]
    [OTHERWISE
        <sentencias del programa>...]
ENDCASE
```

DO WHILE...ENDDO

El formato general de la estructura es:

```
DO WHILE <condición>
    <sentencias del programa>...
ENDDO
```

EDIT [FIELDS <*lista de campos*>] [<*número de registro*>] [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>] [NOINIT] [NOFOLLOW] [NOAPPEND] [NOMENU] [NODELETE] [NOCLEAR]

EJECT

EJECT PAGE

ERASE <*nombre de archivo*>

EXIT

EXPORT TO <*nombre de archivo*> [TYPE DBS/DBASEII/FW2/RPD] [FIELD <*lista de campos*>] [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

FIND <*cadena de caracteres*>/<*n*>

FUNCTION <*nombre de función*>

GO/GOTO BOTTOM/TOP/[RECORD] <*expN*> [IN <*alias*>]

HELP [<*palabra clave*>]

IF...ELSE...ENDIF

El formato general de la estructura es:

```
IF <condición>
   <sentencias del programa>...
[ELSE
   <sentencias del programa...>]
ENDIF
```

IMPORT FROM <*nombre de archivo*> [TYPE] PFS/DBASEII/FW2/ROD/WK1

INDEX ON <*expresión clave*> TO <*nombre de archivo.NDX*> / TAG <*nombre de etiqueta índice .MDX*> [OF <*nombre de archivo .MDX*>] [UNIQUE] [DESCENDING]

INPUT [<*expC*>] TO <*varmemo*>

INSERT [BEFORE]

INSERT BLANK

JOIN WITH <*alis*> TO <*nombre de archivo*> FOR <*condición*> [FIELDS <*lista de campos*>]

LABEL FORM <*archivo de formato de etiqueta*> [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>] [TO PRINTER/TO FILE <*nombre del archivo*>] [SAMPLE]

LIST <*lista exp*>] [<*alcance*>] [OFF] [FOR <*condición*>] [WHILE <*condición*>] [OFF] [TO PRINTER/TO FILE <*nombre de archivo*>] [SAMPLE]

LIST FILES [LIKE <*esquema*>] [TO PRINTER/TO FILE <*nombre de archivo*>]

LIST HISTORY [LAST <*expN*>] [TO PRINTER/TO FILE <*nombre de archivo*>]

LIST MEMORY [TO PRINTER/TO FILE <*nombre de archivo*>]

LIST STATUS [TO PRINTER/TO FILE <*nombre de archivo*>]

LIST STRUCTURE [IN <*alias*>] [TO PRINTER/TO FILE <*nombre de archivo*>]

LIST USERS

LOAD <*nombre de archivo binario*>

LOCATE [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

LOGOUT

LOOP

MODIFY APPLICATION <*nombre del archivo de aplicación*>

MODIFY COMMAND/FILE [<*nombre de archivo*>] [WINDOW <*nombre de ventana*>]

MODIFY LABEL <*nombre de formato de etiqueta*>

MODIFY QUERY/VIEW <*nombre de archivo de consulta*>

MODIFY REPORT <*nombre de formulario de informe*>

MODIFY SCREEN <*nombre de formato de pantalla*>

MODIFY STRUCTURE

MOVE WINDOW <*nombre de ventana*> TO <*fila*>, <*columna*>
/BY <*diferencia entre fila*>, <*diferencia entre columna*>

NOTE/*/&&

ON ERROR/ESCAPE/KEY [LABEL <*nombre de etiqueta clave*>] [<*orden*>]

ON PAD <*nombre de la opción*> OF <*nombre del menú*>
[ACTIVATE POPUP <*nombre popup*>]

ON PAGE [AT LINE <*expN*> <*orden*>]

ON READERROR [<*orden*>]

ON SELECTION PAD <*nombre de opción*> OF <*nombre del menú*> [<*orden*>]

ON SELECTION POPUP <*nombre popup*> / ALL [<*orden*>]

PACK

PARAMETERS <*lista de nombre de parámetro*>

PLAY MACRO <*nombre de macro*>

PRINTJOB...ENDPRINTJOB

PRIVATE <*lista de varmemo*> / ALL[LIKE/EXCEPT <*esquema*>]

PROCEDURE <*nombre de procedimiento*>

PROTECT

PUBLIC <*lista de varmemo*> / [ARRAY <*lista de elementos del array*>]

QUIT

READ [SAVE]

RECALL [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

REINDEX

RELEASE <*lista de varmemo*> / ALL [LIKE/EXCEPT <*esquema*>]

RELEASE MODULE [<*lista del nombre de módulo*>]/MENUS [<*lista del nombre de menú*>]/POPUPS [<*lista del hombre popup*>]/ WINDOWS [<*lista del nombre de ventana*>]

RENAME <*nombre de archivo anterior*> TO <*nombre de archivo nuevo*>

REPLACE [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>] <*campo 1*> WITH <*exp 1*> [ADDITIVE] [, <*campo 2*> WITH <*exp 2*>...]

REPORT FORM <*archivo de formato de informe*> [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>] [TO PRINTER/TO FILE <*nombre de archivo*>][NOEJECT] [PLAIN] [SUMMARY] [HEADING <*expC*>]

RESET [IN <*alias*>]

RESTORE FROM <*nombre de archivo*> [ADDITIVE]

RESTORE MACROS FROM <*archivo de macro*>

RESTORE SCREEN FROM <*nombre de la pantalla*>

RESTORE WINDOW <*lista de nombre de ventana*> / ALL FROM <*nombre de archivo*>

RESUME

RETRY

RETURN [<*expresión*> / TO MASTER / TO <*nombre de procedimiento*>]

ROLLBACK [*nombre de base de datos*]

RUN/! <*orden de DOS*>

SAVE TO <*nombre de archivo*> [ALL [LIKE/EXCEPT <*esquema*>]]

SAVE MACROS TO <*archivo de macro*>

SAVE SCREEN TO <*nombre de pantalla*>

SAVE WINDOW <*lista de nombre de ventana*> / ALL TO <*nombre de archivo de ventana*>

SCAN ... ENDSCAN

El formato general es el siguiente:

```
SCAN [<alcance>] [FOR <condición>] [WHILE <condición>]
   <sentencias del programa>...
ENDSCAN
```

SEEK <*exp*>

SELECT <*área de trabajo*> / <*alias*>

SET

SHOW MENU <*nombre del menú*> [PAD <*nombre de la opción*>]

SHOW POPUP <*nombre popup*>

SKIP [<*expN*>] [IN <*alias*>]

SORT TO <*nombre de archivo nuevo*> ON <*campo 1*> [/A]/[/D] [/C] [, <*campo 2*> [/A]/[/D] [/C]...] [ASCENDING]/[DESCENDING] [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

STORE <*exp*> TO <*lista de varmemo*> / <*lista de elementos de array*> <*varmemo*> / <*elemento de array*> = <*exp*>

SUM [<*lista expN*>] [TO <*lista de varmemo*> / TO ARRAY <*nombre del array*>] [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

SUSPEND

TEXT...ENDTEXT

TOTAL ON <*campo clave*> TO <*nombre de archivo*> [FIELDS <*lista de campos*>] [<*alcance*>] [FOR <*condición*>] [WHILE <*condición*>]

TYPE <*nombre de archivo*> [TO PRINTER/TO FILE <*nombre de archivo*>] [NUMBER]

UNLOCK [ALL/IN <*alias*>]

UPDATE ON <*campo clave*> FROM <*alias*> REPLACE <*campo 1*> WITH <*exp 1*> [, <*campo 2*> WITH <*exp 2*>...] [RANDOM]

USE <*nombre de archivo*> [IN <*área de trabajo*>] [[INDEX <*lista de archivos .MDX y .NDX*>] [ORDER [TAG] <*índice .NDX*> / <*etiqueta índice .MDX*> [OF <*nombre de archivo .MDX*>]]] [ALIAS <*alias*>] [EXCLUSIVE] [NOUPDATE]

WAIT [<*expC*>] [TO <*varmemo*>]

ZAP

SINTAXIS DE LAS ORDENES SET

Nota: Los valores por omisión que aceptan valores numéricos se listan en el Apéndice D con las entradas correspondientes del CONFIG.DB.

SET ALTERNATE on/OFF
SET ALTERNATE TO [<*nombre de archivo*>] [ADDITIVE]

SET AUTOSAVE on/OFF

SET BELL ON/off
SET BELL TO [<*frecuencia*>, <*duración*>]

SET BLOCKSIZE TO <*número*>

SET BORDER TO [SINGLE/double/panel/none/ <*caracteres de contorno*>]

SET CARRY on/OFF
SET CARRY TO [<*lista de campos*> [ADDITIVE]

SET CATALOG on/OFF

SET CATALOG TO [<*nombre del archivo de catálogo*>]

SET CENTURY on/OFF

SET CLOCK on/OFF

SET CLOCK TO [<*fila*>, <*columna*>]

SET COLOR ON/OFF

SET COLOR TO [<*primer plano estándar / fondo estándar*>]
[, <*primer plano realzado/fondo realzado*>] [, <*borde*>]

SET COLOR OF NORMAL/MESSAGES/TITLE/BOX/HIGHLIGHT/
INFORMATION/FIELDS TO [<*primer plano/fondo*>]

SET CONFIRM on/OFF

SET CONSOLE ON/off

SET CURRENCY TO [<*expC*>]

SET CURRENCY LEFT/right

SET DATE [TO] AMERICAN/ansi/british/french/german/italian/japan/usa/
mdy/dmy/ymd

Los formatos son los siguientes:

AMERICANO	*MM/DD/YY*
ANSI	*YY.MM.DD*
BRITANICO	*DD/MM/YY*
FRANCES	*DD/MM/YY*
ALEMAN	*DD.MM.YY*
ITALIANO	*DD-MM-YY*
JAPONES	*YY/MM/DD*
USA	*M-DD-YY*
MDY	*MM/DD/YY*
DMY	*DD/MM/YY*
YMD	*YY/MM/DD*

SET DEBUG on/OFF

SET DECIMALS TO <expN>

SET DEFAULT TO <*unidad*> [:]

SET DELETED on/OFF

SET DELIMITERS on/OFF

SET DELIMITERS TO [<*delimitador(es)*>] [DEFAULT]

SET DESIGN ON/off

SET DEVELOPMENT ON/off

SET DEVICE TO SCREEN/printer/file<*nombre de archivo*>

SET DISPLAY TO MONO/COLOR/EGA25/EGA43/MONO43

SET ECHO on/OFF

SET ENCRYPTION ON/off

SET ESCAPE ON/OFF

SET EXACT on/OFF

SET EXCLUSIVE on/OFF

SET FIELDS on/OFF

SET FIELDS TO <*lista de campos*> / ALL [LIKE/EXCEPT <*esquema*>]

SET FILTER TO [<*condición*>]

SET FORMAT TO [?/ <*nombre de archivo de formato de pantalla*>]

SET FULLPATH on/OFF

SET FUNCTION <*nombre de clave*> / <*número de clave*> TO <*expC*>

SET HEADING ON/off

SET HELP ON/off

SET HISTORY ON/off

SET HISTORY TO <*expN*>

SET HOURS TO 12/24

SET INDEX TO [?/ <*lista de archivos índice*> [ORDER [TAG] <*nombre de índice .NDX*> / <*nombre de etiqueta de índice .MDX*> [OF <*nombre de archivo índice .MDX*>]]]

SET INSTRUCT ON/off

SET INTENSITY ON/off

SET LOCK ON/off

SET MARGIN TO <*expN*>

SET MARK TO [<*carácter*>]

SET MEMOWIDTH TO <*expN*>

SET MESSAGE TO [<*expC*]

SET NEAR on/OFF

SET ODOMETER TO [<*expN*>]

SET ORDER TO [<*expN*> /[TAG] <*archivo índice*>
[OF <*nombre de archivo índice .MDX*>]]

SET PATH TO [<*lista de nombre de camino*>]

SET PAUSE on/OFF

SET POINT TO [<*carácter*>]

SET PRECISION TO [<*expN*>]

SET PRINTER on/OFF

SET PRINTER TO [<*nombre de unidad de DOS*> / FILE <*nombre de archivo*> /
\\SPOOLER/\\CAPTURE]

SET PRINTER TO \\ <*nombre de la computadora*> \ <*nombre de impresora*> =
<*nombre de la unidad de DOS*>

SET PROCEDURE TO [<*nombre de archivo de procedimiento*>]

SET REFRESH TO <*número*>

SET RELATION TO [<*expresión clave*> / <*expN*> INTO <*alias*>
[, <*expresión clave*> / <*expN*> INTO <*alias*>]...]

SET REPROCESS TO <*número*>

SET SAFETY ON/OFF

SET SCOREBOARD ON/off

SET SEPARATOR TO [<*carácter*>]

SET SKIP TO [<*alias1*> [, <*alias*>...]

SET SPACE ON/off

SET SOL on/OFF

SET STATUS ON/off

SET STEP on/OFF

SET TALK ON/off

SET TITLE ON/off

SET TRAP on/OFF

SET TYPEAHEAD TO <*expN*>

SET UNIQUE on/OFF

SET VIEW TO <*nombre de archivo de consulta*>

SET WINDOW OF MEMO TO <*nombre de ventana*>

APENDICE C

Funciones

FUNCIONES POR TIPOS

Matemáticas

ABS	MOD
CEILING	PI
EXP	RAND
FLOOR	ROUND
INT	SIGN
LOG	SQRT
LOG10	

Trigonometría

ACOS	TAN
ASIN	ATN2
COS	SIN
DTOR	TAN
RTOD	

Financial

PV	PAYMENT
NPV	PV

Carácter String y Memo

&	MLINE
AT	REPLICATE
CHR	RIGHT
DIFFERENCE	RTRIM
ISLOWER	SOUNDEX
ISUPPER	SPACE
LEFT	STUFF
LEN	SUBSTR
LIKE	TRANSFORM
LOWER	TRIM
LTRIM	UPPER
MEMLINES	

Fecha y Tiempo

CDOW	DOW
CMONTH	MDY
DATE	MONTH
DAY	TIME
DMY	YEAR

Data Type y Type Conversion

ASC	FLOAT
CTOD	ISALPHA
DTOC	STR
DTOS	TYPE
FIXED	VAL

Data Base

ALIAS	LOOKUP
BOF	LUPDATE
COMPLETED	MDX
DBF	NDX
DELETED	ORDER
EOF	RECCOUNT
FIELD	RECNO
FOUND	RECSIZE
ISMARKED	SEEK
KEY	TAG

Decision-Making

IFF	MIN
MAX	

Enviroment

CALL	FKMAX
COL	GETENV
DATE	INKEY
DISKSPACE	ISCOLOR
ERROR	LATKEY
FILE	LINENO
MEMORY	READKEY
MESSAGE	ROLLBACK
OS	ROW
PCOL	SELECT
PRINTSTATUS	TIME
PROGRAM	VARREAD
PROW	VERSION

Menu y Ventana

BAR	POPUP
MENU	PROMPT
PAD	

Networking y Security

ACCESS	LOCK
CHANGE	NETWORK
FLOCK	BLOCK
LKSYS	USER

FUNCIONES

&<variable de carácter>

ABS(<*expN*>)

Resultado	Valor absoluto de un número
Entrada	Expresión numérica
Salida	Número

ACCESS()

Resultado	Nivel de acceso del usuario
Entrada	Ninguna
Salida	Número

ACOS(<*coseno*>)

Resultado	Arco coseno
Entrada	Expresión numérica
Salida	Número

ALIAS(<*área de trabajo*>)

Resultado	Alias del archivo
Entrada	Expresión numérica o cadena de caracteres (opcional)
Salida	Número

ASC(<*expC*>)

Resultado	Código ASCII decimal del primer carácter de la cadena
Entrada	Expresión de cadena de caracteres
Salida	Número

ASIN(<*seno*>)

Resultado	Arco seno
Entrada	Expresión numérica
Salida	Número el coma flotante (radianes)

AT(<*expC 1*>, <*expC 2*>)

Resultado	Posición inicial de *expC 1* en *expC 2*
Entrada	Expresión de cadena de caracteres, expresión de cadena de caracteres
Salida	Número

ATAN(<*tangente*>)

Resultado	Arco tangente
Entrada	Expresión numérica
Salida	Número en coma flotante (radianes)

ATN2(<*seno*>, <*coseno*>)

Resultado	Arco tangente
Entrada	Expresión numérica, expresión numérica
Salida	Número en coma flotante (radianes)

BAR()

Resultado	Línea destacada en el último pop-up activo
Entrada	Ninguno
Salida	Número

BOF([(<*alias*>])

Resultado	¿Está el puntero de registro al final del archivo?
Entrada	Expresión de carácter o numérica (opcional)
Salida	Valor lógico

CALL(<*programa*> [, <*exp 1*>/<*var 1*> [, <*exp 2*>/<*var 2*>]|...)

Resultado	Valor calculado por un programa binario cargado con LOAD
Entrada	Expresión de cadena de caracteres, hasta siete nombres de expresiones de cadena de caracteres o variables de memoria
Salida	Expresión

CDOW(<*expD*>)

Resultado	Nombre del día
Entrada	Expresión de fecha
Salida	Cadena de caracteres

CEILING(<*expN*>)

Resultado	Entero más pequeño, mayor o igual a un número
Entrada	Expresión numérica
Salida	Número

CHANGE()

Resultado	¿Cambió otro usuario el registro desde la última lectura?
Entrada	Ninguna
Salida	Valor lógico

CHR(<*código ASCII*>)

Resultado	Carácter con un código ASCII decimal especificado
Entrada	Expresión numérica
Salida	Carácter

CMONTH(<*expD*>)

Resultado	Nombre del mes
Entrada	Expresión de fecha
Salida	Cadena de caracteres

COL()

Resultado	Columna actual del cursor
Entrada	Ninguna
Salida	Número

COMPLETED()

Resultado	¿Ha sido completada la última actualización con éxito?
Entrada	Ninguna
Salida	Valor lógico

COS(<*ángulo*>)

Resultado	Coseno de un ángulo
Entrada	Expresión numérica (radianes)
Salida	Número en coma flotante

CTOD(<*expC*>)

Resultado	Fecha correspondiente a una cadena de caracteres
Entrada	Expresión de carácter
Salida	Fecha

DATE()

Resultado	Fecha actual del sistema
Entrada	Ninguna
Salida	Fecha

DAY(<*expD*>)

Resultado	Día del mes
Entrada	Expresión de fecha
Salida	Número

DBF([<*alias*>])

Resultado	Nombre de la base de datos
Entrada	Expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

DELETED([<*alias*>])

Resultado	¿Está marcado el registro actual?
Entrada	Expresión numérica o de carácter (opcional)
Salida	Valor lógico

DIFFERENCE(<*expC 1*>, <*expC 2*>)

Resultado	Diferencia entre los códigos de sonido
Entrada	Expresión de carácter, expresión de carácter
Salida	Número

DISKSPACE()

Resultado	Espacio disponible en el disco en la unidad por omisión
Entrada	Ninguna
Salida	Número

DMY(<*expD*>)

Resultado	Fecha en el formato CO *mes* YY
Entrada	Expresión de fecha.
Salida	Cadena de caracteres

DOW(<*expD*>)

Resultado	Día de la semana
Entrada	Expresión de fecha
Salida	Número

DTOC(<*expD*>)

Resultado	Cadena de caracteres que corresponde a una fecha válida
Entrada	Expresión de fecha
Salida	Cadena de caracteres

DTOR(<*grados*>)

Resultado	Radianes equivalentes a grados
Entrada	Expresión numérica (grados)
Salida	Número

DTOS(<*expD*>)

Resultado	Cadena de caracteres en el formato YYYYMMDD correspondiente a un valor de fecha válido
Entrada	Expresión de fecha
Salida	Cadena de caracteres

EOF([<*alias*>])

Resultado	¿Está el puntero de registro al final del archivo?
Entrada	Expresión de carácter o numérica
Salida	Valor lógico

ERROR()

Resultado	Número de error
Entrada	Ninguna
Salida	Número

EXP(<*exponente*>)

Resultado	e elevado al exponente especificado
Entrada	Expresión numérica
Salida	Número en coma flotante

FIELD(<*número de campo*> [, <*alias*>])

Resultado	Nombre de campo
Entrada	Expresión numérica, expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

FILE(<*nombre de archivo*>)

Resultado	¿Está presente el archivo en el disco?
Entrada	Expresión de carácter
Salida	Valor lógico

FIXED(<*expN*>)

Resultado	Número en coma fija
Entrada	Expresión numérica
Salida	Cadena de caracteres

FKLABEL(<*número de tecla de función*>)

Resultado	Nombre de tecla de función
Entrada	Expresión numérica
Salida	Cadena de caracteres

FKMAX()

Resultado	Número máximo de teclas de función
Entrada	Ninguna
Salida	Número

FLOAT(<*expN*>)

Resultado	Número en coma flotante
Entrada	Expresión numérica
Salida	Número en coma flotante

FLOCK([<*alias*>])

Resultado	¿Puede ser bloqueado el archivo? En caso afirmativo, dBASE IV bloquea el archivo
Entrada	Expresión de carácter ó numérica (opcional)
Salida	Valor lógico

FLOOR(<*expN*>)

Resultado	Mayor entero que es menor o igual a un número
Entrada	Expresión numérica
Salida	Número

FOUND([<*alias*>])

Resultado	¿Se encontró un registro en la búsqueda por clave de índice?
Entrada	Expresión de carácter o numérica (opcional)
Salida	Valor lógico

FV(<*capital*>, <*interés*>, <*periodos*>)

Resultado	Valor futuro de una inversión
Entrada	Expresión numérica, expresión numérica, expresión numérica
Salida	Número

GETENV(<*variable del entorno*>)

Resultado	Valor de la variable del entorno del DOS
Entrada	Expresión de carácter
Salida	Cadena de caracteres

IIF(<*condición*>, <*valor devuelto .T.*>, <*valor devuelto .F.*>)

Resultado	Una de las dos expresiones dependiendo del valor lógico de la condición
Entrada	Expresión lógica, expresión, expresión
Salida	El mismo tipo de datos que el valor devuelto .T. y el valor devuelto .F.

INKEY([<*segundos*>])

Resultado	Número correspondiente a la tecla pulsada por el usuario
Entrada	Número (opcional)
Salida	Número

INT(<*expN*>)

Resultado	Parte entera de un número
Entrada	Expresión numérica
Salida	Número

ISALPHA(<*expC*>)

Resultado	¿Es el primer carácter de una expresión alfabética?
Entrada	Expresión de carácter
Salida	Valor lógico

ISCOLOR()

Resultado	¿Está presente un adaptador de visualización de color?
Entrada	Ninguna
Salida	Valor lógico

ISLOWER(<*expC*>)

Resultado	¿Está el primer carácter de una expresión en mayúscula?
Entrada	Expresión de carácter
Salida	Valor lógico

ISMARKED([<*alias*>])

Resultado	¿Está siendo actualizado el registro actual en una transacción?
Entrada	Expresión numérica o de carácter (opcional)
Salida	Valor lógico

ISUPPER(<*expC*>)

Resultado	¿Está el primer carácter de una expresión en mayúscula?
Entrada	Expresión de carácter
Salida	Valor lógico

KEY([<*archivo índice .MDX*>,] <*número de etiqueta*> [, <*alias*>])

Resultado	Expresión clave del índice
Entrada	Expresión de carácter, expresión numérica, expresión numérica o de carácter
Salida	Cadena de caracteres

LASTKEY()

Resultado	La última tecla pulsada por el usuario
Entrada	Ninguna
Salida	Número

LEFT(<*expC*>, <*longitud*>)

Resultado	Parte de una cadena de caracteres, empezando por la izquierda
Entrada	Expresión de cadena de caracteres, expresión numérica
Salida	Cadena de caracteres

LEN (<*expC*>)

Resultado	Longitud de una cadena de caracteres
Entrada	Expresión de cadena de caracteres
Salida	Número

LIKE(<*esquema*>, <*expC*>)

Resultado	¿Coincide el valor de *expC* con el esquema?
Entrada	Expresión de carácter, expresión de carácter
Salida	Valor lógico

LINENO()

Resultado	Siguiente línea del programa a ejecutarse
Entrada	Ninguna
Salida	Número

LKSYS(<*expN*>)

Resultado	Hora, fecha, o usuario que realiza el último bloqueo
Entrada	Expresión numérica
Salida	Cadena de caracteres

LOCK([<*número de registros*>] [, <*alias*>]) o RLOCK()

Resultado	¿Puede ser bloqueado el registro(s)? En caso afirmativo, dBASE IV bloquea el registro(s)
Entrada	Lisa de expresión numérica como cadena de caracteres, expresión numérica o de carácter (opcional)
Salida	Valor lógico

LOG(<*expN*>)

Resultado	Logaritmo en base e (natural)
Entrada	Expresión numérica
Salida	Número en coma flotante

LOG10(<*expN*>)

Resultado	Logaritmo en base 10
Entrada	Expresión numérica
Salida	Número en coma flotante

LOOKUP(<*campo devuelto*>, <*expresión de búsqueda*>, <*nombre de campo*>)

Resultado	Campo localizado en otra base de datos
Entrada	Expresión, expresión, nombre de campo
Salida	Expresión

LOWER(<*expC*>)

Resultado	Cadena de caracteres en minúscula
Entrada	Expresión de cadena de caracteres
Salida	Cadena de caracteres

LTRIM(<*expC*>)

Resultado	Cadena de caracteres sin los espacios en blanco de la izquierda
Entrada	Expresión de cadena de caracteres
Salida	Cadena de caracteres

LUPDATE([<*alias*>])

Resultado	Fecha de la última actualización
Entrada	Expresión numérica o de carácter (opcional)
Salida	Fecha

MAX(<*exp 1*>, <*exp 2*>)

Resultado	El máximo de dos valores
Entrada	Expresión, expresión
Salida	Número, fecha, o cadena de caracteres

MDX(<*expN*>/[, <*alias*>])

Resultado	Nombre de archivo .MDX
Entrada	Expresión numérica, expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

MDY(<*expD*>)

Resultado	Fecha en formato MM/DD/YY
Entrada	Expresión de fecha
Salida	Cadena de caracteres

MEMLINES(<*nombre de campo memo*>)

Resultado	Número de líneas en el campo memo en la anchura actual
Entrada	Expresión de cadena de caracteres
Salida	Número

MEMORY()

Resultado	Memoria disponible
Entrada	Ninguna
Salida	Número

MENU()

Resultado	Nombre del último menú de líneas activo
Entrada	Ninguna
Salida	Cadena de caracteres

MESSAGE()

Resultado	Texto del mensaje de error
Entrada	Expresión numérica, expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

MIN(<*exp 1*>, <*exp 2*>)

Resultado	El mínimo de dos valores
Entrada	Expresión, expresión
Salida	Número, fecha, o cadena de caracteres

MLINE(<*nombre de campo memo*>, <*número de línea*>)

Resultado	Una línea primera de un campo memo
Entrada	Expresión cadena de caracteres, expresión numérica
Salida	Cadena de caracteres

MOD(<*expN 1*>, <*expN 2*>)

Resultado	Módulo (resto de una división)
Entrada	Expresión numérica, expresión numérica
Salida	Número

MONTH(<*expD*>)

Resultado	Mes
Entrada	Expresión de fecha
Salida	Número

.MDX(<*número de índice*>|[, <*alias*>])

Resultado	Nombre de índice .NDX
Entrada	Expresión numérica, expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

NEXTWORK()

Resultado	¿Está ejecutándose dBASE IV en una red?
Entrada	Ninguna
Salida	Valor lógico

ORDER([<*alias*>])

Resultado	Nombre del índice maestro
Entrada	Expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

OS()

Resultado	Nombre del sistema operativo
Entrada	Ninguna
Salida	Cadena de caracteres

PAD()

Resultado	Nombre de la última opción de la línea de menús destacada
Entrada	Ninguna
Salida	Cadena de caracteres

PAYMENT(<*principal*>, <*interés*>, <*periodos*>)

Resultado	Cantidad pagada en un préstamo
Entrada	Expresión numérica, expresión numérica, expresión numérica
Salida	Número

PCOL()

Resultado	La coordenada de la columna actual de la impresora
Entrada	Ninguna
Salida	Número

PI()

Resultado	Pi
Entrada	Ninguna
Salida	Número en coma flotante

POPUP()

Resultado	Nombre del último menú de ventana activo
Entrada	Ninguna
Salida	Cadena de caracteres

PRINTSTATUS()

Resultado	¿Está preparada la impresora para recibir los datos?
Entrada	Ninguna
Salida	Valor lógico

PROGRAM()

Resultado	Nombre del procedimiento (o programa) que se ejecuta actualmente
Entrada	Ninguna
Salida	Cadena de caracteres

PROMPT()

Resultado	Texto visualizado en la opción o línea de menús destacada
Entrada	Ninguna
Salida	Cadena de caracteres

PROW()

Resultado	La coordenada de la fila actual de la impresora
Entrada	Ninguna
Salida	Número

PV(<*pago*>, <*interés*>, <*períodos*>)

Resultado	Valor actual de una inversión
Entrada	Expresión numérica, expresión numérica, expresión numérica
Salida	Número

RAND([<*expN*>])

Resultado	Número aleatorio
Entrada	Expresión numérica (opcional)
Salida	Número

READKEY()

Resultado	Número correspondiente a la tecla pulsada al salir de la última READ
Entrada	Ninguna
Salida	Número

RECCOUNT([<*alias*>])

Resultado	Contador de registro (número de registros en el archivo)
Entrada	Expresión numérica o cadena de caracteres (opcional)
Salida	Número

RECNO ([<*alias*>])

Resultado	Número del registro actual
Entrada	Expresión numérica o de cadena de caracteres (opcional)
Salida	Número

RECSIZE([<*alias*>])

Resultado	Tamaño del registro (longitud total del registro)
Entrada	Expresión numérica o de cadena de caracteres (opcional)
Salida	Número

REPLICATE(<*expC*>, <*repeticiones*>)

Resultado	Una cadena de caracteres repetida
Entrada	Expresión de cadena de caracteres, expresión numérica
Salida	Cadena de caracteres

RIGHT(<*expC*>, <*expN*>)

Resultado	Parte de una cadena de caracteres, empezando por la derecha
Entrada	Expresión de cadena de caracteres, expresión numérica
Salida	Cadena de caracteres

RLOCK([<*lista de expresiones*>] [<*alias*>]) o LOCK( )

Resultado	¿Puede ser bloqueado el registro(s)? En caso afirmativo, dBASE IV bloquea el registro(s)
Entrada	Lista de expresiones numéricas tal como cadena de caracteres, expresión numérica o de carácter (opcional)
Salida	Valor lógico

ROLLBACK()

Resultado	¿Se recuperó la transacción con éxito?
Entrada	Ninguna
Salida	Valor lógico

ROUND(<*expN*>, <*posiciones decimales*>)

Resultado	Número redondeado al número de posiciones decimales especificada
Entrada	Expresión numérica, expresión numérica
Salida	Número

ROW()

Resultado	Coordenada de la fila actual del cursor
Entrada	Ninguna
Salida	Número

RTOD(<*expN*>)

Resultado	Grados correspondientes a radianes
Entrada	Expresión numérica (radianes)
Salida	Número

RTRIM(<*expC*>)

Resultado	Cadena de caracteres sin los espacios en blanco del final
Entrada	Expresión de carácter
Salida	Cadena de caracteres

SEEK(<*exp*>[, <*alias*>])

Resultado	¿Encontró SEEK el registro coincidente?
Entrada	Expresión, expresión numérica o de carácter (opcional)
Salida	Valor lógico

SELECT()

Resultado	Area de trabajo disponible con mayor numeración
Entrada	Ninguna
Salida	Número

SET(<*expC*>)

Resultado	Valor de un parámetro del entorno de dBASE IV
Entrada	Expresión de carácter
Salida	Cadena de caracteres

SIGN(<*expN*>)

Resultado	Signo de un número
Entrada	Expresión numérica
Salida	Número

SIN(<*ángulo*>)

Resultado	Seno de un ángulo
Entrada	Expresión numérica (radianes)
Salida	Número en coma flotante

SOUNDEX(<*expC*>)

Resultado	Código de sonido
Entrada	Expresión de cadena de caracteres
Salida	Cadena de caracteres

SPACE(<*longitud*>)

Resultado	Cadena de espacios en blanco
Entrada	Expresión numérica
Salida	Cadena de caracteres

SQRT (<*expN*>)

Resultado	Raíz cuadrada de un número
Entrada	Expresión numérica
Salida	Número en coma flotante

STR(<*expN*>[, <*longitud*>] [, <*decimales*>])

Resultado	Representación en cadena de caracteres de un número
Entrada	Expresión numérica, expresión numérica, expresión numérica
Salida	Cadena de caracteres

STUFF(<*expC 1*>, <*posición inicial*>, <*longitud*>, <*sustitución*>)

Resultado	Cadena de caracteres modificada
Entrada	Expresión de cadena de caracteres, expresión numérica, expresión numérica, expresión de cadena de caracteres
Salida	Cadena de caracteres

SUBSTR(<*expC*>, <*posición inicial*> [, <*longitud*>])

Resultado	Subcadena (parte de una cadena de caracteres)
Entrada	Expresión de cadena de caracteres, expresión numérica, expresión numérica
Salida	Cadena de caracteres

TAG([<*archivo índice .MDX*>,] <*número de etiqueta*> [, <*alias*>])

Resultado	Nombre de etiqueta
Entrada	Expresión numérica, expresión numérica o de carácter (opcional)
Salida	Cadena de caracteres

TAN(<*ángulo*>)

Resultado	Tangente de un ángulo
Entrada	Expresión numérica (radianes)
Salida	Número en coma flotante

TIME( )

Resultado	Hora actual del sistema
Entrada	Ninguna
Salida	Cadena de caracteres

TRANSFORM(<*exp*>, <*imagen*>)

Resultado	Cadena de caracteres formateada
Entrada	Expresión
Salida	Cadena de caracteres

TRIM(<*expC*>)

Resultado	Cadena de caracteres sin espacios en blanco
Entrada	Expresión de cadena de caracteres
Salida	Cadena de caracteres

TYPE(<*expC*>)

Resultado	Tipo de datos de una expresión
Entrada	Expresión de carácter
Salida	Carácter

UPPER(<*expC*>)

Resultado	Cadena de caracteres en mayúscula
Entrada	Expresión de cadena de caracteres
Salida	Cadena de caracteres

USER()

Resultado	Nombre de identificación del usuario actual de la red
Entrada	Ninguna
Salida	Cadena de caracteres

VAL(<*expC*>)

Resultado	Valor numérico correspondiente a una cadena de caracteres
Entrada	Expresión de cadena de caracteres
Salida	Número

VARREAD()

Resultado	Nombre de la variable captada con @...GET
Entrada	Ninguna
Salida	Cadena de caracteres

VERSION()

Resultado	Versión de dBASE IV
Entrada	Ninguna
Salida	Cadena de caracteres

YEAR(<*fecha*>)

Resultado	Año
Entrada	Expresión de fecha
Salida	Número