di Francesco Petroni

Visual Basic Intermedio Griglie ed affini

Iniziamo con questo una serie di articoli dedicati agli utilizzatori del Visual Basic, articoli che avranno un "taglio intermedio": sono pensati per una precisa categoria di utilizzatori, non quelli alle prime armi né quelli già esperti, ma quelli che stanno in mezzo, che hanno già le conoscenze di base e che vogliono approfondire alcuni temi particolari.

Gli esperti invece già dovrebbero conoscere gli argomenti trattati e quindi dovrebbero anche essere in grado di svolgere facilmente gli esercizi che corredano l'articolo... ma non è detto.

Ogni articolo della serie tratterà uno specifico argomento, ad esempio questo parla di Griglie ed affini, il prossimo parlerà di Oggetti e Collezioni di Oggetti, nel successivo articolo l'argomento sarà Applicazioni SDI, MDI e Explorer Style, eccetera.

Prima parte

Alcune premesse

Gli articoli saranno prevalentemente pratici, nel senso che proporranno una serie di esercizi facilmente rieseguibili da ciascuno di voi. In particolare in questo primo articolo ogni esercizio corrisponde ad una figura che mostra sia il listato sia, in primo piano, la Form che costituisce aspetto esteriore dell'applicazione.

Il fatto che gli articoli siano destinati a lettori già pratici del Visual Basic ci permette di limitare la descrizione del listato (comunque pubblicato tutto) al minimo indispensabile e di concentrarci di più sugli aspetti generali relativi agli oggetti che si stanno trattando.

Diamo per note le conoscenza delle modalità operative con le quali si crea, si esegue, si compila, un programma Visual Basic, diamo per nota la conoscenza dei componenti, quelli standard e quelli che si caricano da libreria (ActiveX, DLL, OCX e sinonimi vari) e della loro modalità di utilizzo in una nostra applicazione. Diamo infine per noti i principi di base della programmazione, i comandi principali del linguaggio VB, l'uso delle variabili, le funzioni, ecc.

Per gli esercizi che prevedono l'accesso ai dati usiamo un database in formato MDB (si chiama **Dati.mdb**) in cui c'è una tabella **Persone**, con dei campi di semplice individuazione. Potete ricrearla oppure modificare i listati per adattarli a dati vostri.

La TextBox, la ListBox, la Grid

Un normale controllo **TextBox** permette di vedere un solo dato per volta. Una altrettanto normale **ListBox** permette di vedere un semplice elen-

co di dati. Una **Griglia** permette di vedere una matrice di dati, organizzati per riga e per colonna e visualizzati per riga e per colonna.

Per riportare all'interno di una TextBox (supponiamo che si chiami T) una stringa basta impostarne la proprietà Text:

T.Text = "Ciao mondo"

Il valore assunto dal componente T è inequivocabilmente "Ciao mondo".

Per riportare all'interno di una **ListBox** (supponiamo che si chiami L) una serie di stringhe, occorre eseguire più volte il metodo AddItem:

L.Additem "ROMA"
L.Additem "MILANO"

Il valore assunto dal componente L dipende dall'elemento selezionato. Per selezionare un elemento o per individuare quale elemento sia quello selezionato si usa la proprietà ListIndex:

L.ListIndex =3

punta il quarto (si parte da 0) elemento e conseguentemente L assume il valore del quarto elemento. Altra proprietà importantissima (è di sola lettura) è quella che indica quanti sono i valori della Lista:

L.ListCount

La griglia, supponiamo di aver caricato il componente aggiuntivo **MSFlexGrid** e di averla chiamata G, si comporta, grosso modo, come una lista a più colonne. Esistono una serie di proprietà che servono per impostare il numero di righe e di colonne (Rows e Cols) ed una serie di proprietà che servono per puntare una data cella o per sapere quale cella sia stata selezionata con il mouse (Row e Col).

Il valore assunto dal componente G dipende dalla cella selezionata, o con il mouse o con le proprietà prima descritte.

Esistono due modi per alimentare una griglia. Il primo è praticabile quando già si conosce la dimensione della griglia e consiste nel selezionare da programma la cella e nell'assegnarli un valore:

G.Row = 12
G.Col = 3
G = "Milano"

Il secondo va utilizzato quando si accodano via via i dati:

C = Chr(9)
G.Cols = 2

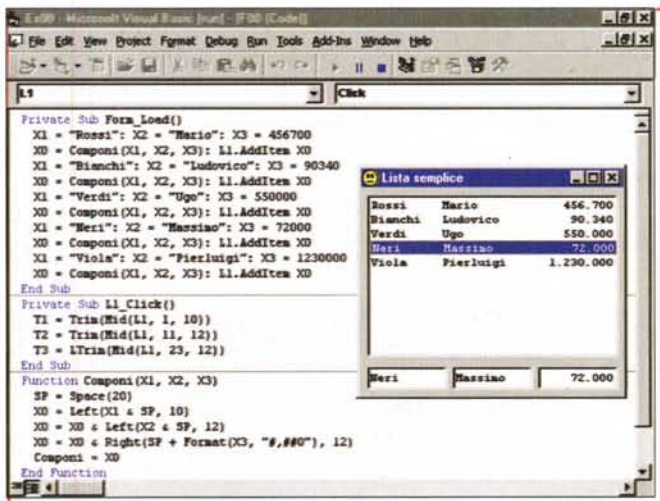


Figura 1 - Visual Basic 5.0 - Una Lista a colonne.
Una TextBox visualizza un dato alla volta, una ListBox una semplice colonna di dati, una Griglia un elenco, organizzato per righe e per colonne, di dati. I primi due controlli fanno parte della dotazione di base del VB, mentre il terzo (in realtà di griglie ce ne sono ben tre) corrisponde ad un ActiveX che va caricato nel programma. Nel primo esercizio vediamo come simulare un incolonnamento di tre dati in una unica ListBox. Il programma risultante è più leggero di quello che usa la Griglia, ma costa un po' di più in termini di programmazione. Per realizzare le "finte" colonne usiamo funzioni di stringa.

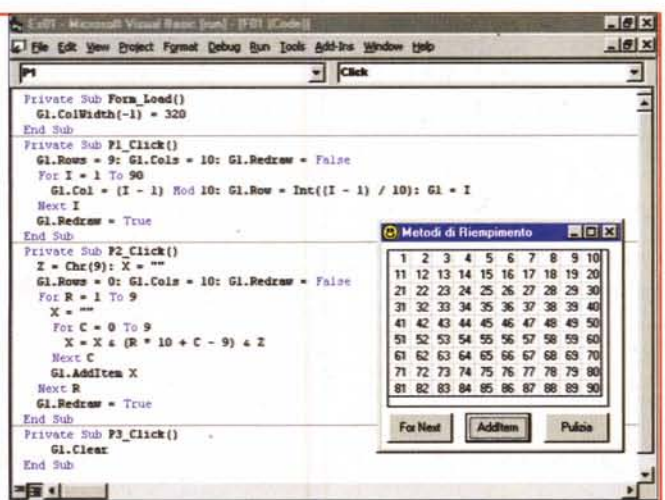


Figura 2 - Visual Basic 5.0 - Due liste usate per scegliere.
Tra le proprietà di una lista ce ne sono due che riguardano la possibilità di eseguire selezione multiple. La prima è MultiSelect, che permette tre varianti (no selezione multipla, selezione semplice e selezione estesa), e la seconda è Style, che prevede l'uso delle CheckBox per selezionare gli elementi. Se si sceglie lo Style con le CheckBox la proprietà MultiSelect deve essere posta uguale a 0. Qui vediamo sia un esempio di uso di due Liste con le CheckBox, sia come sia possibile leggere i valori selezionati, grazie alla proprietà, di tipo logico, Selected(Index).

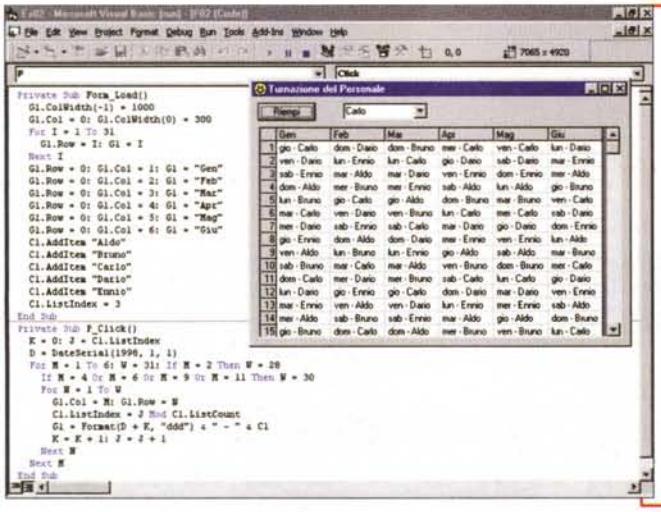


Figura 3 - Visual Basic 5.0 - I due modi per riempire una FlexGrid.
Per i nostri esercizi sulle griglie useremo la MSFlexGrid, che nel Visual Basic 5.0 sostituisce la più rudimentale MSGrid, presente nel Visual Basic 4.0 e precedenti. Nel testo descriveremo con maggior approfondimento le differenze tra le due griglie ora citate e la terza, che si chiama DataBound e che è la più adatta per lavorare sui Recordset provenienti dai Database. La FlexGrid si può alimentare in due maniere, definendone le dimensioni in termini di righe e di colonne e riempiendo le celle così definite, oppure sfruttando il metodo Additem, che accoda le celle e quindi crea le righe. Se i dati da caricare sono ordinati conviene usare questo secondo sistema. Nell'esercizio vediamo i due modi usati per caricare una serie di numeri che vanno da 1 a 90.

G.Rows = 1
G.Additem "Roma" & C & "Rutelli"
G.Additem "Napoli" & C & "Bassolino"

Additem aggiunge una riga, il carattere 9 (tabulazione) serve per spostarsi sulla colonna a fianco.

I primi tre esercizi

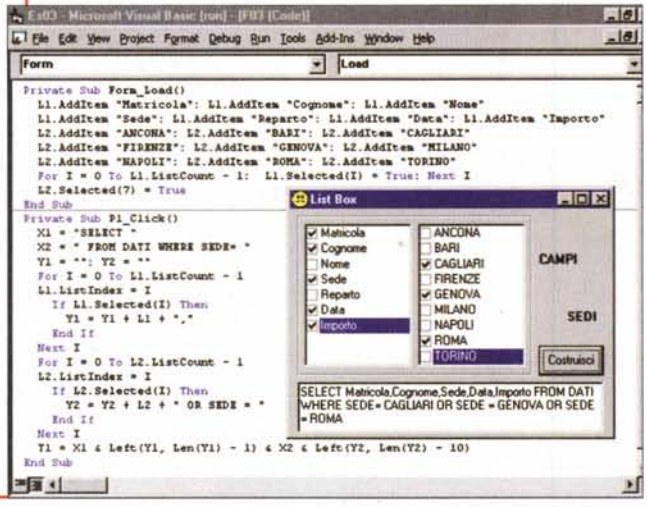
Il primo esercizio (**figura 1**), ben descritto nella didascalia, suggerisce come simulare un incolonnamento in una semplice ListBox. Se occorre caricare solo poche colonne, se si vuole realizzare un programma leggero, può essere opportuno utilizzare un controllo standard anziché uno che va caricato.

Perché il sistema possa funzionare occorre scegliere, per la lista, un Font non proporzionale, ad esempio il classico CourierNew (anzi diciamo che dovete usare il CourierNew).

Il secondo esercizio (**figura 2**) mostra una serie di varianti d'uso della Lista, che ne ampliano di un bel po' l'ambito di utilizzo. Quando la lista permette la selezione multipla è necessario esplorare elemento per elemento il valore della proprietà Selected(Index) per individuare se quell'elemento è selezionato o meno.

Il terzo esercizio (**figura 3**) esegue lo "slalom parallelo" sui due modi di caricare la griglia, quello basato sul puntamento e quello basato sull'Additem.

Figura 4 - Visual Basic 5.0 - Una Griglia coinvolta in un processo di calcolo.
Questo esercizio non è una novità. Usiamo una Griglia per creare ed alimentare un nostro Calendario nel quale segnare i turni di servizio del personale di una ditta in cui si lavora anche nei festivi (ad esempio il turno di una Guardia Notturna). Nelle celle scriviamo il giorno della settimana (mentre mese e giorno del mese appaiono nella prima colonna e nella prima riga) e il nome del... malcapitato di turno. Dal punto di vista programmazione occorre contare i giorni successivi alla data di partenza e valutare con un semplice stratagemma numerico "a chi tocca" fare il turno.



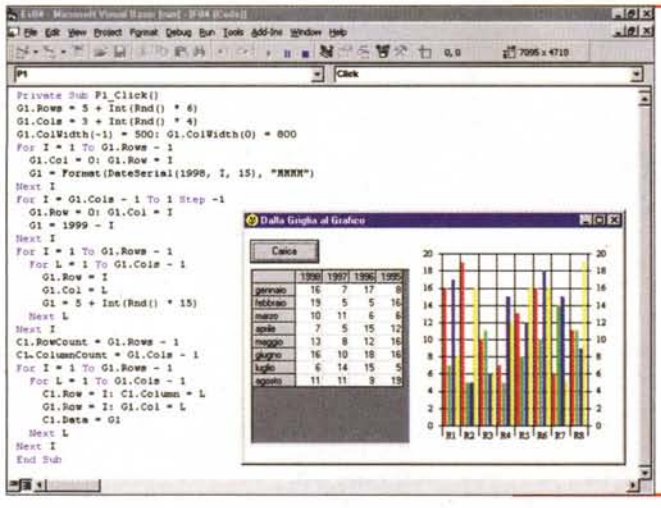


Figura 5 - Visual Basic 5.0 - Estrazione dei numeri della tombola.
Nel terzo esercizio, quello in figura 3, abbiamo studiato due modi per alimentare la griglia (puntamento della cella ed accodamento), fregandocene un po' di quello che stavamo mettendo nella griglia stessa. In questo esercizio, usando il metodo del puntamento della riga e della colonna, riempiamo la griglia con i numeri della tombola estraendoli a caso da una matrice di numeri. L'aspetto più interessante dell'esercizio è proprio questo: l'estrazione senza ripetizione di un elemento da una serie di elementi (problematica tipica dei giochi di carte).

Piccola storia delle Griglie

La prima griglia è apparsa con il Visual Basic 3.0, si chiamava **MSGrid** e permetteva i due metodi di caricamento ora descritti. Il Visual Basic 4.0 ha conservato la MSGrid ed ha introdotto il **DataBound Grid Control**, una griglia adatta ad essere collegata ad un

DataControl per visualizzare e per gestire i suoi dati. Il Visual Basic 5.0 ha introdotto la **MSFlexGrid**, in sostituzione della MSGrid, ed ha confermato al suo posto la DataBound.

Alcune brevi precisazioni:

- per garantire la compatibilità con il programmi realizzati con le versioni precedenti, nel CD del VB5 è tuttora disponibile la MSGrid (è un pò nascosta ma c'è);
- la Flex e la DataBound possono essere utilizzate in modalità "bound" ed in modalità "unbound". Bound significa che la griglia prende i dati direttamente da un DataControl, Unbound significa che i dati vanno caricati "a mano";
- la griglia Flex non consente l'editazione delle celle, lavora solo in output. La DataBound invece permette l'editazione delle celle e l'aggiunta, da tastiera, di nuove righe;
- sia la Flex che la DataBound sono versioni ridotte di componenti ActiveX, realizzate da case software indipendenti (la VideoSoft e la Apex) che propongono versioni più complete dei loro prodotti. Per gli interessati una visitina ai due siti può fornire utili indicazioni e versioni trial.

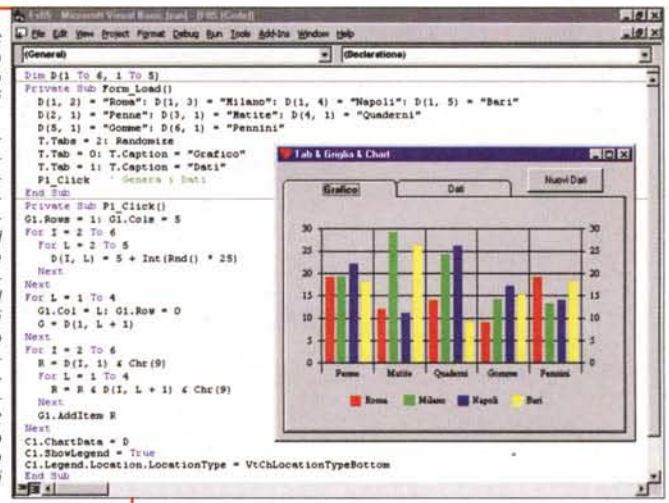


Figura 6 - Visual Basic 5.0 - Il controllo MSChart è il compagno ideale per una griglia di numeri.
Un insieme di dati numerici può essere memorizzato in una Matrice di N righe per M colonne oppure in una Griglia, che corrisponde ad una matrice rettangolare solo che si "vede" fisicamente. Non è raro il caso in cui l'insieme di dati numerici debbono essere usati per realizzare un grafico. Il caricamento dei dati necessari all'oggetto MSChart può essere eseguito con lo stesso sistema del puntamento che si usa per le Griglie.

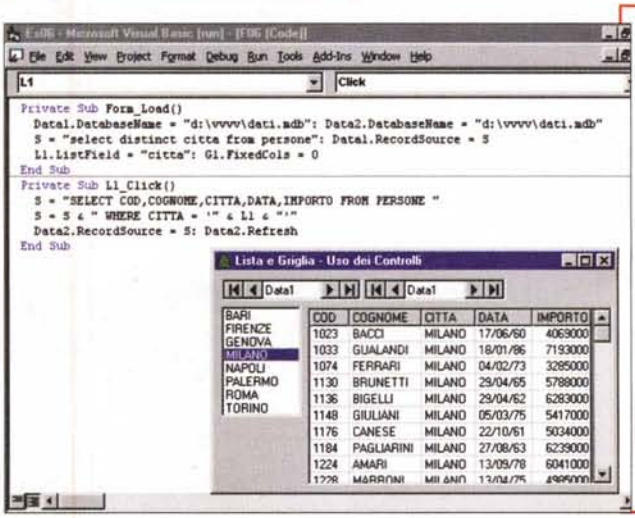


Figura 7 - Visual Basic 5.0 - Variante con uso di un controllo SSTab e di una Matrice.
In questa variante dell'esercizio precedente vediamo alcune cosette in più. La prima è l'utilizzo del Controllo SSTab che permette di separare in due pagine, identificate dalle linguette, la Griglia con i dati dal Diagramma che li mostra in forma di Istogramma. La seconda è la gestione della Legenda, gestibile come tutti gli altri componenti del diagramma, da programma. La terza è la possibilità di alimentare il diagramma direttamente con una matrice di dati numerici.

Altri quattro esercizi

Il quarto esercizio (**figura 4**) dimostra la flessibilità della griglia che è adatta agli utilizzi più disparati. La usiamo per creare un piccolo calendario.

Il quinto esercizio (**figura 5**), che in un certo senso vi dovevamo, ci serve per ribadire il fatto che la griglia è solo un modo per visualizzare i nostri dati. Ad esempio se inventiamo un sistema per estrarre i numeri della tombola questo prescinde dal fatto che poi, per comodità, vediamo i numeri stessi nella griglia.

La griglia è un contenitore di dati, organizzati per riga e per colonna. I tutti i linguag-

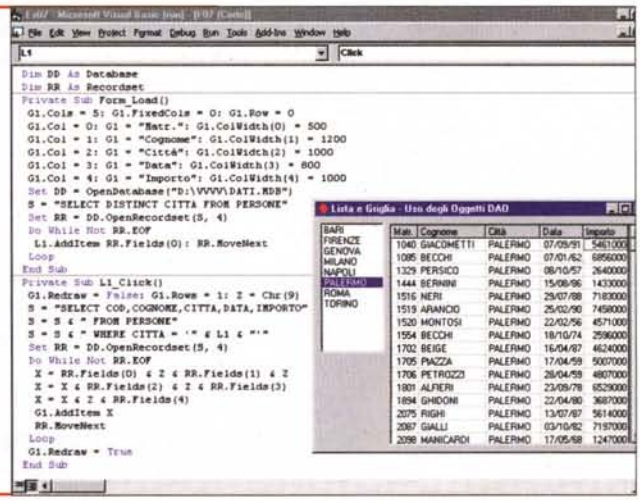


Figura 8 - Visual Basic 5.0 - Alle prese con un Database - Uso dei DataControls.
In quasi tutte le applicazioni i controlli presenti nelle Form vengono utilizzati per visualizzare i dati provenienti da un Database e quindi da oggetti di tipo RecordSet (tabelle, dynaset e snaposhot). Il sistema più semplice per agganciare un RecordSet è l'utilizzo del controllo Data, detto anche DataControl, per il quale vanno impostate una serie di proprietà fondamentali: Connett, che specifica il tipo di database agganciato, DatabaseName, dal significato intuibile, e RecordSource, che può essere sia il nome di una tabella che una espressione SQL. E' anche importante specificare, a seconda dell'uso che si farà dei dati, il tipo di RecordSet. Gli altri oggetti in cui visualizzare i dati (nel nostro caso una DBList ed una MSFlexGrid) vanno agganciati al DataControl.

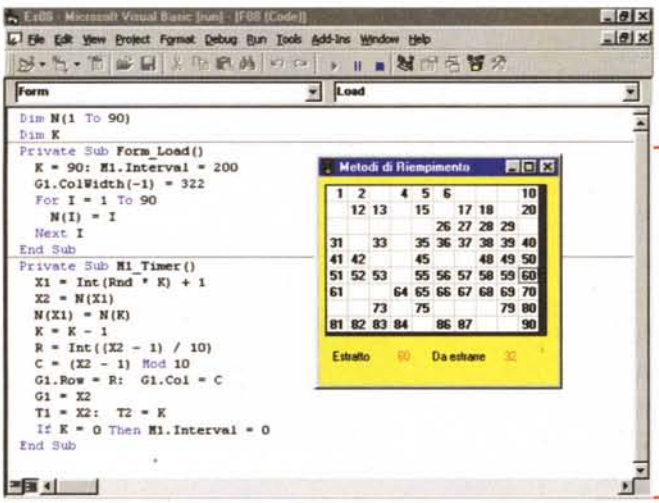


Figura 9 - Visual Basic 5.0 - Alle prese con un Database - Uso degli oggetti DAO.
Nell'esercizio precedente abbiamo visto come alimentare con due DataControls un oggetto DBList e un oggetto MSFlexGrid. In questo esercizio vediamo come evitare di usare il DataControl mediante la programmazione degli oggetti DAO. Evitiamo anche di usare la DBList che sostituiamo con una più leggera ListBox, mentre la MSFlexGrid la alimentiamo con un semplice metodo Additem. Si programma di più, ma l'applicazione è più leggera.

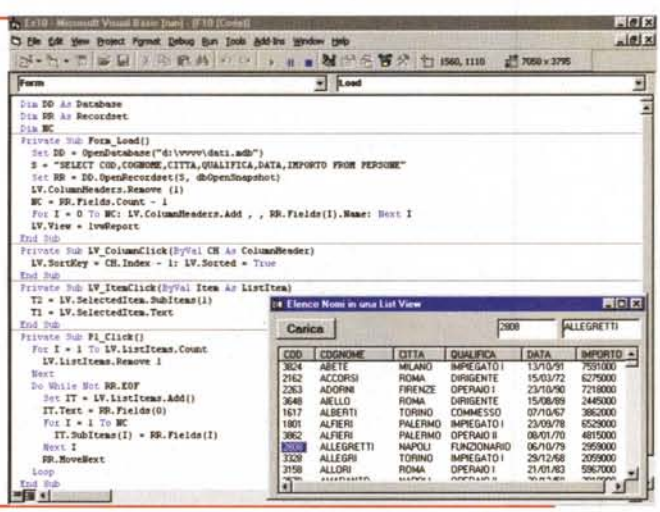


Figura 10 - Visual Basic 5.0 – Un'elaborazione Batch per alimentare una griglia.
Utilizziamo la stessa tabella vista nell'esercizio precedente sulla quale faremo una semplice elaborazione statistica (sulla tabella Persone) per contare il numero di persone appartenenti ad una Città e ad una Qualifica. Anche l'elenco delle città e delle qualifiche, con il quale costruiamo la prima riga e la prima colonna della Griglia con i risultati, sarà ottenuto elaborando la stessa tabella Persone. Scegliamo la strada più lunga (ce la saremmo potuta cavare con una query di raggruppamento) scorrendo tutti i record e identificando la cella in cui aggiornare il contatore.

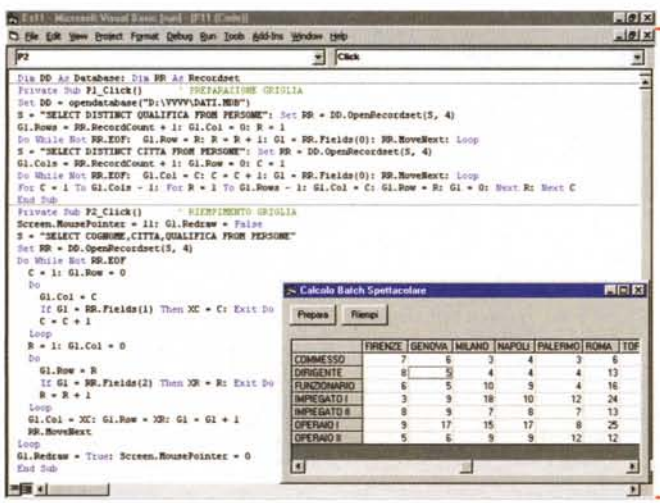


Figura 11 - Visual Basic 5.0 - Il DataBound Grid Control: quando serve gestire i dati.
Anche la DataBound Grid va associata ad un DataControl. Dispone di una serie di proprietà che vanno opportunamente impostate nel caso, ad esempio, si voglia permette la modifica dei dati e l'accodamento di un nuovo record. Le possibili varianti di utilizzo sono tantissime, ve ne proponiamo una che comporta la visualizzazione, in sei caselle di testo, di sei campi della Tabella. Queste caselle di testo le usiamo sia per visualizzare i dati della persona selezionata sia per alimentare la tabella con un nuovo record.

gi di programmazione esiste la Matrice che è, a sua volta, un modo per memorizzare, in questa variabile speciale, i dati. Griglia e Matrice sono quindi parenti stretti.

Uno degli usi più frequenti di dati organizzati per riga e per colonna è nella realizzazione dei grafici.

Il componente ActiveX **MSChart**, di cui vediamo due esempi applicativi (**figura 6** ed in **figura 7**), può appoggiarsi facilmente sia una matrice (quando non occorre vedere i dati numerici) che ad una griglia.

Alla prese con i dati

L'utilizzo più frequente e più nobile della griglia consiste nella visualizzazione dei dati di un database.

Il sistema più semplice passa attraverso l'uso di un controllo **Data**, per il quale si impostano DatabaseName e RecordSource (una tabella o un'espressione SQL), e che va indicato come DataSource della griglia. In pratica, piazzando nella form un DataControl ed una MSFlexGrid, basta impostare tre proprietà per vedere i dati nella griglia.

La griglia si può anche caricare da programma, ad esempio se il database viene letto mediante la manipolazione degli oggetti **DAO**. Costa di più in termini di programmazione ma si evita l'utilizzo del DataControl, che peraltro non da fastidio a nessuno e si può anche rendere invisibile.

Altri tre esercizi

Vi proponiamo due varianti dello stesso esercizio: selezione di una Città e produzione di un elenco di Persone di quella Città. Nel primo caso (**figura 8**) usiamo due DataControls, una DataBound List (che

viene alimentata dal primo DataControl) ed una FlexGrid. Se si sceglie una Città nella lista (evento Click) viene modificata la proprietà RecordSource del secondo DataControl e quindi riempita con i nuovi dati la griglia.

Nel secondo caso non usiamo i DataControl né la DBList, che sostituiamo con una semplice ListBox. La Flex, così come la ListBox, la alimentiamo da programma maneggiando gli oggetti DAO. Il programma (**figura 9**) risulta un pò più complesso.

In **figura 10** vediamo invece un esempio di calcolo pesante ed inutile (nel senso che lo stesso risultato si poteva raggiungere in un modo più diretto). Creiamo una griglia con le Città come intestazioni di Colonna e Qualifiche come intestazioni di Riga. Scorrendo la tabella individuiamo la Città e la Qualifica del singolo record e aggiorniamo il contatore.

Il modo più diretto consiste nel fare una Query di tipo CrossTab, di passarla come RecordSource ad un DataControl che a sua volta passa i dati alla griglia.

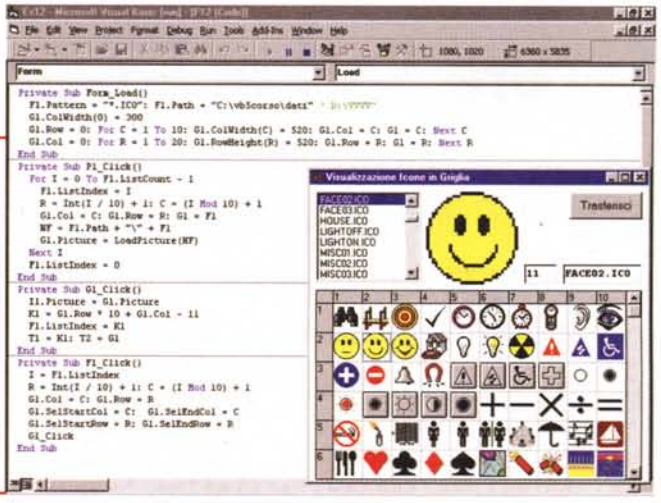


Figura 12 - Visual Basic 5.0 - La vecchia MSGrid.
L'ottima MSFlexGrid, che è un po' la reginetta della festa (anche perché è uno degli oggetti con più ambiti di applicazione) sostituisce la vecchia MSGrid che è vissuta fino alla versione 4.0 del VisualBasic. La vecchia MSGrid (che è comunque disponibile anche nel CD di VB5, per garantire la compatibilità delle vecchie applicazioni) funzionava grosso modo come quella nuova. Non permette un accesso diretto ai dati ma permette, come quella nuova, grazie alla classica proprietà Picture, anche di inserire delle immagini nelle celle.

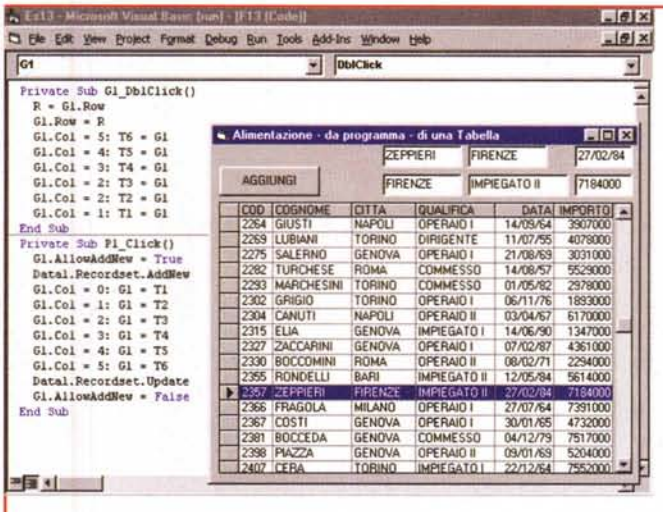


Figura 13 - Visual Basic 5.0 - Una ListView è meglio di una griglia (in certi casi). Windows 95 ci ha abituati ad una serie di nuovi oggetti che, chi vuole sviluppare applicazioni rispettando le modalità standard di interazione tra applicazione stessa ed utente, deve saper utilizzare. Tutti gli oggetti nuovi di Windows 95 sono contenuti nella libreria OCX che si chiama Microsoft Windows Common Controls 5.0. Tra questi troviamo tutti gli oggetti che siamo abituati ad usare, come la ListView che permette di vedere i dati in una sorta di superlista, dimensionabile, cliccabile, ecc. ecc.

Le altre griglie e gli esercizi finali

La differenza fondamentale tra la MSFlexGrid e la DataBound Grid consiste nel fatto che le celle della prima non sono digitabili. La **DataBound Grid** si può associare ad un DataControl e le sue celle sono editabili. Questo significa che con la DataBound Grid è possibile manipolare, e quindi non solo visualizzare, ma anche modificare, eliminare, aggiungere record in un Database. Questa possibilità amplifica enormemente gli ambiti di utilizzo dell'oggetto. Ne diamo una piccola esemplificazioni in **figura 11**.

Una delle possibilità che aveva la MSGrid e che viene conservata nella nuova FlexGrid è quella di "ospitare" nelle sue celle un file grafico, che si affianca e non sostituisce l'eventuale contenuto testuale. Le istruzioni sono, per la MSGrid:

G1.Picture = LoadPicture(<nome del file>)

La MSFlexGrid permette sia di caricare immagini nelle celle, con la seguente istruzione:

Set G1.CellPicture = LoadPicture(<nome del file>)

sia si caricare immagini che occupano più celle.

Vediamo, in **figura 12**, l'esempio di quanto detto.

Chiudiamo mostrando un altro oggetto, la **ListView**, che può sostituirsi alla griglia quando occorra visualizzare dei dati organizzati per riga e per colonna. La ListView è il classico oggetto "contenitore" di Windows 95 che permette quattro modalità di visualizzazione (Icone Grandi, Icone Piccole, Elenco e Dettagli).

La modalità dettagli è quella più adatta ai nostri dati. Nell'ultimo esercizio, che vi proponiamo in **figura 13**, vediamo come caricare gli Item della lista e come attivare le funzionalità di ordinamento della lista stessa al click sul titolo della colonna.

MC